Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Urša Juvan

Iskanje ponavljajočih tem in vzorcev v glasbi s pomočjo kompozicionalnega hierarhičnega modela

magistrsko delo

študijski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana, 2016

Hvala ...

... mentorju doc. dr. Matiji Maroltu za hitre in podrobne popravke končne naloge in vmesnih predstavitev, dosegljivost po mailu in na govorilnih urah ter posredovanje pri birokratskih zapletih.

... asistentu Matevžu Pesku za predlog teme, deljenje znanja, idej in nasvetov, usmerjanje in pomoč pri razvoju, vse popoldanske in sobotne ure, preživete v laboratoriju, stalno dosegljivost po mailu, Skypu in telefonu, za vse kave, za pregled vmesnih predstavitev in končne naloge, za psihično podporo in spodbudo.

... Mihu za odstop prostora in pripravo delovnega okolja za razvoj na strežniku, pomoč in podporo tekom celotnega magistrskega študija ter izdatno pomoč v času zaključevanja magistrske naloge.

... staršem za podporo tekom celotnega študija.

... mami, Špeli, očetu, Branki, Neži, Gregu, Borisu, babici in Lauri za varstvo in ostalo pomoč.

... Špeli za lekturo.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CHM	Compositional Hierarchical Model	Kompozicionalni hierarhični model
SymCHM	Symbolic Compositional Hierarchical Model	Kompozicionalni hierarhični model za simbolni glasbeni zapis
MIR	Music Information Retrieval	pridobivanje informacij iz glasbe
ISMIR	International Society for Music Information Retrieval	mednarodna skupnost za pridobivanje informacij iz glasbe
MIREX	Music Information Retrieval Evaluation eXchange	dogodek za skupno evalvacijo rezultatov s področja pridobivanja informacij iz glasbe
MIDI	Musical Instrument Digital Interface	glasbeni instrumentalni digitalni vmesnik

Povzetek

Naslov: Iskanje ponavljajočih tem in vzorcev v glasbi s pomočjo kompozicionalnega hierarhičnega modela

V magistrski nalogi se posvetimo iskanju ponavljajočih vzorcev v glasbi, kar spada na področje pridobivanja informacij iz glasbe. Uporabimo kompozicionalni hierarhični model, ki je bil izdelan za reševanje drugih nalog s tega področja, in ga prilagodimo za obdelavo skladb v simbolnem glasbenem zapisu ter iskanje ponavljajočih vzorcev. Za lažje testiranje in interpretacijo rezultatov implementiramo tudi vizualizacijo modela. Rezultate evalviramo na dveh zbirkah podatkov in jih primerjamo z rezultati ostalih raziskovalcev. Primerjava pokaže, da kompozicionalni hierarhični model enako dobro ali bolje kot ostali algoritmi najde veliko število ponovitev prepoznanega vzorca, nekoliko pa zaostaja v iskanju vsaj ene pojavitve čim več glasbeno relevantnih vzorcev. Za izboljšave šibkejših točk modela predlagamo več možnih rešitev ter podamo možnosti za njegov nadaljnji razvoj.

Ključne besede: Pridobivanje informacij iz glasbe, iskanje vzorcev, kompozicionalni hierarhični model, simbolni glasbeni zapis.

Abstract

Title: Discovery of Repeated Themes and Sections in Music with a Compositional Hierarchical Model

In this thesis we focus on the discovery of repeated patterns in music, one of the tasks in the research field of music information retrieval. We use a compositional hierarchical model, which was previously successfully applied to other tasks from this field, and adjust it to process symbolic music and find repeated patterns in musical pieces. In order to facilitate testing and interpretation of algorithm's output we develop a visual representation of the model. Results are evaluated on two datasets and compared to results of other researchers. Comparison shows that the proposed model is comparable to or better than other approaches in finding multiple repetitions of one pattern, but lags in identifying at least one occurrence of each structurally salient pattern. We propose several possible solutions for increasing the share of successfully identified salient patterns and suggest some guidelines for further development of the model.

Keywords: Music information retrieval, pattern discovery, compositional hierarchical model, symbolic music representation.

Poglavje 1

Uvod

Pridobivanje informacij iz glasbe (angl. *Music Information Retrieval* – MIR) je interdisciplinarno raziskovalno področje, ki združuje znanja iz muzikologije, računalništva, psihologije, informatike, strojnega učenja, procesiranja signalov in drugih področij. Obsega različne naloge, kot je na primer avtomatska harmonizacija, ugotavljanje plagiatorstva oziroma iskanje podobnosti v skladbah, avtomatska klasifikacija glasbe, prepoznavanje akordov ali inštrumentov, avtomatska transkripcija in drugo. Glavna prednost računalnikov v povezavi z glasbo je zmožnost hitre obdelave podatkov, ki lahko znatno olajša delo glasbenemu ekspertu in omogoči navigacijo po velikih glasbenih zbirkah. Ena od muzikoloških nalog, pri kateri lahko izkoristimo prednosti računalnika, je tudi glasbena analiza.

Glasbena analiza je muzikološki proces, s katerim analitik poskuša razložiti skladateljevo kompozicijo in poslušalčevo dojemanje nekega glasbenega dela. Izvede se lahko na enem glasbenem delu, na posameznem odseku skladbe ali na zbirki del. Pogosto vključuje razčlenitev skladb na več manjših gradnikov, preučevanje njihovih soodvisnosti in povezovanje gradnikov v večje smiselne celote. Pri tem imajo posebno vlogo gradniki, ki se v izbranem delu ali preko več del večkrat ponovijo. Huron ponavljanje označi kot "*osnovni princip strukturiranja glasbe v mnogih glasbenih stilih. Poslušalca vodi skozi doživljanje in olajša proces pomnjenja glasbe*" [20].

Avtomatsko iskanje ponavljajočih vzorcev igra pomembno vlogo v muzikoloških študijah, poleg tega pa lahko doprinese tudi k boljšemu razumevanju skladateljeve kompozicije in poslušalčeve interpretacije osnovne strukture glasbe. Računalniški pristopi lahko znatno olajšajo analizo posamezne skladbe, prav tako pa tudi celotnih zbirk skladb, v katerih lahko odkrijejo zanimive relacije med vzorci preko več skladb. Druge možne uporabe avtomatizacije so izboljšana navigacija po velikih zbirkah skladb ali obsežnih samostojnih skladbah, prav tako pa tudi v računalniško podprtih orodjih za kompozicijo glasbe [41].

V magistrski nalogi želimo razviti orodje za avtomatsko iskanje ponavljajočih vzorcev preko ene ali več skladb. Za ta namen želimo uporabiti kompozicionalni hierarhični model [43], ki je bil do sedaj uporabljen za uspešno reševanje nalog prepoznavanja akordov in osnovnih frekvenc v skladbah, podanih v avdio zapisu, ter ga prilagoditi za iskanje ponavljajočih vzorcev v skladbah, podanih v simbolnem zapisu.

Model s hierarhično strukturo modelira človekovo dojemanje glasbe. Posamezne elemente glasbe predstavi na več nivojih kompleksnosti, kjer so gradniki na posameznih nivojih sestavljeni kot kompozicije gradnikov z nižjih nivojev – tako kot se v glasbi več frekvenc avdio signala združuje v en ton, toni v akorde in akordi v harmonije. Analogno hierarhično organizacijo glasbe opazujemo v strukturiranju glasbe skozi ponavljajoče vzorce, kjer se posamezni toni združujejo v glasbene motive, motivi pa v glasbene teme. Transparentnost modela omogoča vpogled v informacije na vseh nivojih modela, kar doprinese k njegovi uporabnosti za analizo glasbe, saj lahko dolge ponavljajoče vzorce opazujemo po posameznih gradnikih. Značilnost modela, ki jo želimo izkoristiti pri iskanju ponavljajočih vzorcev, je tudi njegova robustnost, ki je posledica relativnosti posameznih gradnikov (elemente glasbe predstavi kot koncept namesto absolutnih vrednosti) in dveh biološko navdahnjenih mehanizmov, halucinacije in inhibicije. Halucinacija omogoči prepoznavo pojavitve vzorca v vhodnih podatkih, ki se vzorcu ne prilegajo popolnoma, inhibicija pa omogoča prečiščevanje hipotez, saj odstranjuje re-

dundantne pojavitve vzorcev. Robustnost modela lahko s pridom izkoristimo za iskanje transponiranih vzorcev (vzorcev, ki se pojavijo z enakomernim zamikom v tonskih višinah glede na originalni vzorec) in variacij (ponovitve se od osnovnega vzorca rahlo razlikujejo).

V drugem poglavju pričnemo s pregledom področja magistrske naloge. Opišemo področje pridobivanja informacij iz glasbe, nalogo iskanja ponavljajočih vzorcev in pristope ostalih raziskovalcev ter predstavimo hierarhično modeliranje glasbe. V tretjem poglavju opišemo strukturo kompozicionalnega hierarhičnega modela za simbolni glasbeni zapis, učenje, pridobivanje rezultata iz testnih podatkov in filtriranje vzorcev za izbor najbolj relevantnih. Predstavimo implementacijo in vizualizacijo modela ter opišemo postopek testiranja. V četrtem poglavju predstavimo testne zbirke skladb in mere, uporabljene za evalvacijo. Opišemo postopek iskanja najboljših parametrov. Podamo rezultate modela na testnih skladbah, jih ovrednotimo in primerjamo z rezultati ostalih raziskovalcev. V petem poglavju strnjeno pregledamo opravljeno delo in dobljene rezultate ter zaključimo z usmeritvami za nadaljnji razvoj.

Poglavje 2

Pregled področja

2.1 Pridobivanje informacij iz glasbe

Pridobivanje informacij iz glasbe (angl. *Music Information Retrieval* – MIR) je interdisciplinarno raziskovalno področje, ki združuje znanja iz muzikologije, računalništva, psihologije, informatike, strojnega učenja, procesiranja signalov in drugih področij. Obsega različne naloge, kot je na primer avtomatska harmonizacija, ugotavljanje plagiatorstva oziroma iskanje podobnosti v skladbah, avtomatska klasifikacija glasbe, prepoznavanje akordov ali inštrumentov, avtomatska transkripcija in drugo.

Naloge v grobem delimo v tri skupine [4]:

- *pridobivanje informacij*, pri čemer na podlagi povpraševanja iščemo ustrezno skladbo (na primer priporočilni sistem na podlagi preteklih nakupov ali iskanje skladbe na podlagi kratkega zvočnega posnetka)
- *klasifikacija*, kjer vhodnim podatkom pripišemo eno pripadajočo vrednost oziroma oznako (na primer identifikacija skladatelja ali ocena tempa)
- *označevanje odsekov*, pri čemer pripišemo več oznak različnim mestom v vhodnih podatkih (na primer prepoznava zaporedja akordov v avdio posnetku)

Vhodni podatki za naloge s tega področja so običajno podani v eni od naslednjih štirih oblik [4]:

- *slike* notnega zapisa v tiskani obliki ali rokopisu
- *simbolni glasbeni zapis*
- *digitalni avdio zapis*
- *metapodatki* (na primer različne kategorije, v katere razvrščamo določeno skladbo, podatki o skladatelju itd.)

Večina sodobnih raziskav se osredotoča na vhodne podatke v avdio zapisu (95 %), ostale pa v večji meri na simbolni glasbeni zapis [4].

2.1.1 Kratka zgodovina MIR

Povzeto po [4].

Z večanjem dostopnosti računalnikov je v 60-ih in 70-ih letih 20. stoletja raslo tudi zanimanje za analizo glasbe s pomočjo računalnika. Izraz *pridobivanje informacij iz glasbe* je bil prvič uporabljen leta 1966 v Kasslerjevem članku z naslovom "Naproti pridobivanju informacij iz glasbe"[24]. Veliko raziskav se je v tem času osredotočalo na optimizacijo simbolnega glasbenega zapisa za računalnik [32], začela pa se je razvijati tudi analiza glasbenega avdio zapisa – med drugim analiza barve zvoka (angl. *timbre*) [48], sledenje tonski višini (angl. *pitch tracking*) [39], prepoznavanje tonske višine, tonalitete, taktovskega načina in tempa (angl. *pitch, key, meter and tempo extraction*) [6] ter prepoznavanje ritma (angl. *rhythm extraction*) [47].

Razvoj računalniških pristopov k MIR se je v 80-ih letih nekoliko upočasnil zaradi pomanjkanja velikih digitalnih glasbenih zbirk, povečalo pa se je zanimanje za kognitivni vidik človekovega dojemanja glasbe. V 90-ih letih se je z večanjem razpoložljivosti digitalnega avdio zapisa in zmogljivosti računalnikov zopet povečalo tudi število raziskav na področju MIR. V tem obdobju sta bila prvič predstavljena povpraševanje z mrmranjem (angl. *query*

by humming) [23] in iskanje v bazah na podlagi avdio zapisa (angl. *search via audio content*) [53].

Konec stoletja se je odvilo nekaj delavnic s tematiko pridobivanja informacij iz glasbe, ki so navdahnile ustanovitev mednarodne skupnosti za pridobivanje informacij iz glasbe (ISMIR, angl. *International Society for Music Information Retrieval*) in vsakoletne istoimenske konference, ki jo le-ta organizira. Konferenca, ki je bila prvič izvedena leta 2000, je postala primarno prizorišče za predstavitev rezultatov novih raziskav na tem področju.

S širitvijo MIR se je pojavila potreba po formaliziranju različnih nalog s tega področja in zagotovitvijo testnih baz in mer za evalvacijo rezultatov, na podlagi katerih bi bilo mogoče ovrednotiti nove raziskave. Za ta namen je bil leta 2005 zasnovan dogodek MIREX, predstavljen v naslednjem poglavju.

Trenutni trend na področju MIR deli raziskave v dve glavni veji: raziskave nižjenivojskih nalog, ki so potrebne za uspešno procesiranje avdio signala (na primer zaznavanje časa nastopa tona (angl. *audio onset detection*), ocenjevanje osnovnih frekvenc (angl. *multiple fundamental frequency estimation*), sledenje ritmu (angl. *audio beat tracking*)) in raziskave višjenivojskih nalog, ki so zanimive z muzikološkega in kulturološkega vidika (na primer ocenjevanje akordov (angl. *chord estimation*), iskanje ponavljajočih vzorcev (angl. *discovery of repeated themes and sections*), strukturna segmentacija glasbe (angl. *structural segmentation*)).

2.1.2 MIREX

MIREX (angl. *Music Information Retrieval Evaluation eXchange*) je dogodek za skupno evalvacijo rezultatov s področja pridobivanja informacij iz glasbe, ki ga organizira Laboratorij za evalvacijo sistemov za pridobivanje informacij iz glasbe (IMIRSEL – angl. *International Music Information Retrieval Systems Evaluation Laboratory*) Univerze v Illinoisu. Dogodek, ki poteka enkrat letno od leta 2005 naprej, raziskovalcem omogoča nepristransko vrednotenje novih modelov. IMIRSEL v okviru dogodka MIREX formalizira naloge s področja pridobivanja informacij iz glasbe, oblikuje testne

zbirke podatkov za posamezne naloge in oblikuje postopke ter mere za evalvacijo rezultatov. Zaradi relevantnosti rezultatov testne zbirke podatkov niso objavljene. Algoritme, ki sodelujejo na dogodku, zato na testnih podatkih poganja organizator, ki nato tudi objavi rezultate. [15]

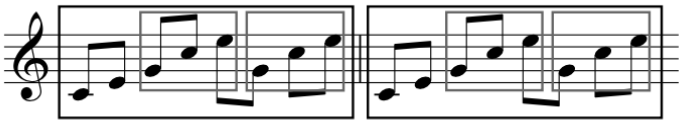
2.2 Iskanje vzorcev v glasbi

Za ponavljajoči glasbeni vzorec ni enotne definicije, saj ga avtorji študij za različne namene in različne glasbene zvrsti formalizirajo na različne načine (npr. samo ponavljanje tonskih višin, samo ponavljanje notnih trajanj, kombinacija tonske višine in trajanja itd.). Zanimajo jih različne dimenzije glasbe: ritem, melodija, dinamika, barva in druge ter njihove kombinacije. Obstaja več nivojev abstrakcije, na katerih predstavijo te dimenzije: kot številske vrednosti, kot kategorije oziroma razrede, kot smer spremembe itd. [9, 21].

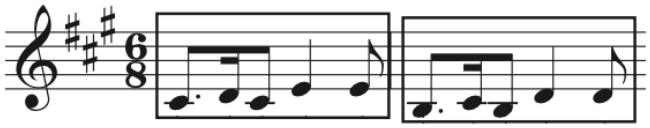
Za potrebe magistrskega dela uporabimo definicijo naloge iskanja ponavljajočih vzorcev, kot je opredeljena v [10]. Vzorec je definiran kot zaporedje tonskih višin s pripadajočimi časi nastopa posameznega tona, ki se v skladbi ponovi vsaj dvakrat. Druga, tretja in nadaljnje pojavitve vzorca se ponavadi pojavijo z nekim časovnim zamikom glede na prvo, lahko so tudi transponirane in delno spremenjene (so variacije osnovnega vzorca in torej niso nujno enake dolžine kot osnovni vzorec). Vzorci se lahko med seboj prekrivajo ali pa so v celoti vsebovani v drugem vzorcu.

Na Slikah 2.1, 2.2 in 2.3 so prikazani primeri ponavljajočih vzorcev. Na Sliki 2.1 sta prikazana dva različna vzorca, za vsakega so ponovitve vzorca popolnoma identične osnovnemu vzorcu. Na Sliki 2.2 je ponovitev transponirana glede na osnovni vzorec, na Sliki 2.3 pa so ponovitvi dodani okraski, spremenjen je tudi ritem vzorca.

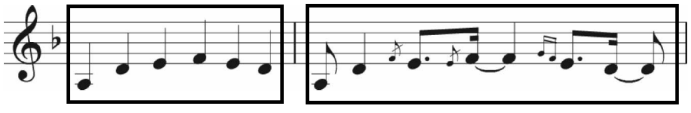
Problem iskanja vzorcev se deli na 4 podprobleme glede na tip vhodnih podatkov: vhodna skladba je lahko podana v avdio ali simbolnem zapisu, v



Slika 2.1: Primer vzorca, kjer je ponovitev identična osnovnemu vzorcu. Na sliki sta prikazana dva vzorca. Prvi vzorec ima dve pojavitvi, ki ju označujeta večja pravokotnika, drugi vzorec, ki je vsebovan v prvem, pa označujejo manjši pravokotniki in ima štiri pojavitve [28].



Slika 2.2: Primer transponiranega vzorca [28].



Slika 2.3: Primer vzorca, kjer ima ponovitev dodane okraske in je ritem nekoliko spremenjen (ponovitev je variacija osnovnega vzorca) [28].

vsakem od zapisov pa je lahko podana monofonično[1] ali polifonično[2]. Polifonično skladbo se lahko za nalogo iskanja vzorcev pretvori v monofonično. Če so glasovi v večglasju označeni (angl. *voiced polyphony*), za pretvorbo vsak glas zapišemo zaporedno enega za drugim. Če skladba ni razdeljena na posamezne glasove (angl. *unvoiced polyphony*), jo je treba najprej smiselno razdeliti na glasove (angl. *voice separation*), da se lahko poslužimo enakega postopka. Za razdelitev na glasove ni generalnega pravila, zato predstavlja eno od aktualnih nalog, ki jo računalniki rešujejo v povezavi z glasbo [5, 7, 34].

Problem iskanja vzorcev ločimo na dva podproblema glede na obseg – iskanje znotraj ene skladbe (angl. *intra-opus discovery*) [50, 29, 38, 41] ali iskanje preko več skladb (*inter-opus discovery*) [22, 14]. Iskanje znotraj ene skladbe je uporabno za analizo posamezne skladbe, iskanje po več skladbah pa za študijo sloga posameznega skladatelja, iskanje plagiatov ali študije zbirk narodnih pesmi z nekega območja.

Metode v grobem delimo v dva razreda glede na pristop k reševanju problema: metode, zasnovane na iskanju vzorcev v nizih podatkov, in geometrične metode. Pri prvih se avtorji osredotočijo na iskanje ponavljanj v podatkih, podanih kot nizi oznak, na primer niz tonskih višin, podanih z imeni tonov (A; G; A; D), ali kot niz tonskih višin, podanih z MIDI številko tona (57; 55; 57; 50), ali kot pari tonske višine in notnega trajanja ((A; 1,5); (G; 0,5); (A; 1,0); (D; 1,0)) [21]. Geometrične metode podatke predstavijo kot večdimenzionalno množico in iščejo podobnosti med posameznimi dimenzijami točk. Ponavljajoče vzorce identificirajo kot identične ali zelo podobne oblike v večdimenzionalnem prostoru [21]. Po raziskavi Mereditha geometrične metode dajejo boljše rezultate za polifonične skladbe kot metode, zasnovane na iskanju vzorcev v nizih podatkov [38].

V magistrski nalogi bomo predstavili reševanje problema z monofoničnimi vhodnimi skladbami v simbolnem glasbenem zapisu. Uporabili bomo pristop z iskanjem vzorcev v nizih podatkov.

[1] Hkrati v skladbi nastopa samo en ton.

[2] V skladbi istočasno zveni več tonov.

Poleg iskanja natančnega ujemanja nas pri ponavljajočih vzorcih zanima tudi delno ujemanje. Lahko se pojavijo variacije v ritmu, melodiji ali katerem drugem elementu skladbe, kot so na primer trilčki, pospeševanje ali upočasnjevanje, spremembe v višini nekaterih tonov in transpozicija [21]. Ta približna ponavljanja mora algoritem zaznati kljub manjšim spremembam. Trenutni pristopi sledijo dvema različnima strategijama. Prva temelji na numerični podobnosti: ponovitev je zabeležena, ko razlika med ponovitvijo in osnovnim vzorcem ne preseže neke mejne vrednosti. Ta mejna vrednost je poljubno določena s strani raziskovalca, zato je relevantnost rezultatov, pridobljenih s to metodo, vprašljiva. Druga strategija predpostavlja, da prepoznava ponovitve vzorca sledi iz prepoznave podobnosti v posameznih dimenzijah glasbe, kot so višina tona, intervali, ritem itd., in njihovih kombinacijah [28].

Računalniški algoritem praviloma najde veliko več ponavljanj v glasbi, kot jih zazna poslušalec ali jih kot pomembne strukturne elemente označi glasbeni analitik. Število vzorcev je pogosto zmanjšano na podlagi statističnih pojavitev vzorcev. Tak način filtriranja pogosto ne izboljša relevantnosti rezultatov, saj lahko izloči nekatere za poslušalca zanimive vzorce [28]. Za razvoj algoritma, ki je sposoben izmed vseh vzorcev izločiti tiste, ki so strukturno pomembni, moramo znati formalno definirati, kaj loči zaznana ponavljanja od tistih, ki jih poslušalec presliši oziroma jih analitik smatra kot nepomembne [38].

Nekatere študije za evalvacijo rezultatov svojih algoritmov uporabijo skladbe z anotiranimi ponavljajočimi vzorci, za katere bomo uporabili izraz “ekspertna anotacija”. Anotacijo običajno prispevajo glasbeni strokovnjaki ali pa vzorce celo označijo skladatelji sami. Če se rezultati algoritma dobro ujemajo z ekspertno anotacijo, smatramo, da algoritem dobro modelira človeško presojo o pomembnosti vzorcev. Algoritme lahko smatramo kot modele človekovega doživljanja glasbe in skozi uspehe in neuspehe modela bolje razumemo, na katerih kriterijih temelji človeška presoja ponavljajočih vzorcev [21].

Sorodni nalogi s področja pridobivanja informacij iz glasbe, ki jih ne smemo zamenjevati z iskanjem ponavljajočih vzorcev, sta iskanje ujemajočih vzorcev (angl. *Pattern Matching*) in avtomatska segmentacija glasbe (angl. *Structural Segmentation Task*). Iskanje ujemajočih vzorcev se nanaša na iskanje ponovitev vnaprej podanih vzorcev v neki skladbi. To se od iskanja ponavljajočih vzorcev razlikuje v tem, da pri iskanju vzorcev nimamo vnaprej podanih vzorcev, ampak jih mora algoritem sam prepoznati in nato najti njegove ponovitve [21]. Pri segmentaciji celotno skladbo razdelimo na vsebinsko zaokrožene dele, ki se med seboj ne prekrivajo. Vzorec pa je del skladbe, ki se večkrat ponovi in se lahko prekriva z drugimi vzorci. Vsi vzorci ne pokrijejo celotne skladbe. [52].

2.3 Sorodna dela

Pristopi različnih skupin raziskovalcev k problemu iskanja ponavljajočih vzorcev v glasbi se med seboj zelo razlikujejo. Eden od razlogov za to so vhodni podatki v različnih oblikah (avdio, simbolni) in različne vrste ponavljanj, ki jih iščejo (samo v višinah tonov, višine in notna trajanja itd.). Nekateri uporabljajo tehnike za iskanje ujemajočih vzorcev in avtomatsko segmentacijo glasbe ter jih prilagodijo za iskanje ponavljajočih vzorcev. V nadaljevanju opisujemo nekaj pristopov, predstavljenih v zadnjih letih.

Metoda, ki jo uporabijo **G. Velarde, T. Weyde in D. Meredith** [50, 51], temelji na ideji, da je ob dobri razdelitvi skladbe na segmente mogoče grupirati podobne segmente v skupine in oceniti njihovo pomembnost znotraj skladbe [51]. Predstavijo algoritem, ki išče vzorce v monofoničnih skladbah, podanih v simbolnem zapisu.

Za reševanje problema uporabijo pristope s področja procesiranja signalov. Skladbo predstavijo kot enodimenzionalen diskreten signal tonske višine v času. Signal je vzorčen iz MIDI datoteke s frekvenco vzorčenja, določeno v vzorcih na četrtinko. Za vsako časovno vrednost dobijo pripadajočo kro-

matično tonsko višino[3]. Za zapis pavz uporabijo dve različni varianti: zapišejo jih kot tonsko višino 0 ali pa jih nadomestijo s tonsko višino, ki sledi pavzi (za pavze na začetku skladbe), oziroma s predhodno tonsko višino.

Signal tonske višine v času $v(t)$ z uporabo valčne transformacije preoblikujejo v vektor koeficientov v času z uporabo analizne funkcije

$$\Psi_{s,u}(t) = \frac{1}{\sqrt{s}}\Psi\left(\frac{t-u}{s}\right), \tag{2.1}$$

kjer je $s > 0$ faktor skaliranja, u pomik v času in $\Psi(t)$ Haarov valček[4] (angl. *Haar wavelet*).

Koeficienti so izračunani kot produkt signala $v(t)$ z analizno funkcijo. Produkt analizne funkcije s signalom $v(t)$ izračuna razliko med povprečno tonsko višino v prvi in drugi polovici izbranega intervala. Koeficienti torej nakazujejo smer gibanja melodije.

Dobljene koeficiente uporabijo za razdelitev skladbe na segmente tako, da postavijo meje segmentov v lokalnih maksimumih in ničlah. Privzeti mejni točki sta tudi začetek in konec skladbe. Ničle v koeficientih se pojavijo tam, kjer je povprečna višina tona v prvi polovici intervala enaka povprečni višini tona v drugi polovici. Lokalne maksimume dobijo, ko se pojavi največja sprememba v povprečni višini tona.

Da dobijo segmente, invariantne na transpozicijo, vsak segment normalizirajo tako, da vrednosti v signalu delijo s povprečno višino tona za ta segment.

Dobljene segmente grupirajo glede na podobnost z ostalimi segmenti. Uporabijo metodo k najbližjih sosedov (angl. *K-nearest neighbours*), kjer kot razdaljo med primeri uporabijo enkrat evklidsko razdaljo, drugič pa manhattansko. Boljše rezultate dobijo z uporabo evklidske razdalje.

Avtorji rezultate opisanega algoritma evalvirajo na dveh različnih zbirkah

[3] MIDI številsko predstavitev višine tona (angl. *MIDI note number*)

[4] $\Psi(t) = \begin{cases} 1 & \text{če } 0 \leq t < 1/2 \\ -1 & \text{če } 1/2 \leq t < 1 \\ 0 & \text{sicer} \end{cases}$

podatkov: Bachovih invencijah (angl. *Bach's Two-Part Inventions (BWV 772–786)*) in na množici 360 nizozemskih narodnih pesmi. Na testnih zbirkah dosežejo do 85-% klasifikacijsko točnost.

Lartillot [29, 28] prestavi algoritem PatMinr, ki išče vzorce v monofoničnih skladbah v simbolnem zapisu z inkrementalnim iskanjem po večdimenzionalnem prostoru parametrov. Skladbo zapiše kot večdimenzionalno množico, v kateri vsaka dimenzija vsebuje podatek bodisi o spremembi melodije (diatonični interval[5], kromatični interval[6], gibanje melodije (navzgor, navzdol, brez premika)) bodisi o spremembi ritma (časovni interval, merjen v številu dob) med dvema zaporednima tonoma. Pri identifikaciji vzorca išče ponovitve v posameznih dimenzijah. Ponovitve vzorca, ki niso popolnoma enake osnovnemu vzorcu (so variacije osnovnega vzorca), se ne ujemajo v vseh dimenzijah.

Vzorci so predstavljeni kot verige stanj, imenovane verige vzorcev (angl. *pattern chains*), ki predstavljajo zaporedje parametrov, ki tvorijo vzorec. Verige vzorcev so združene v drevo vzorcev, kjer vsaka veja predstavlja eno verigo vzorcev.

Algoritem se skozi celoten niz podatkov sprehodi samo enkrat in sproti gradi drevo vzorcev. Če algoritem za neko zaporedje podatkov najde ustrezno vejo v drevesu in temu zaporedju sledi nov podatek, ki v drevesu še ne obstaja v tem zaporedju, se doda novo vozlišče na nižjem nivoju.

Lartillot rezultate algoritma na štirih različnih skladbah predstavi zgolj kvalitativno.

Meredith idr. [38] prvi uporabijo geometrično metodo reševanja problema. Predstavijo skupino algoritmov za iskanje vzorcev v polifonih skladbah v simbolnem zapisu.

Glasbo zapišejo kot večdimenzionalno množico D, kjer prva dimenzija

[5]Diatonični interval je razlika med stopnjami tonov v tonaliteti skladbe. Če imamo na primer skladbo v A duru, ton A predstavlja stopnjo 0, ton H stopnjo 1 itd.

[6]razlika v MIDI številki tona

predstavlja čas nastopa tona v številu šestnajstink od začetka skladbe, druga dimenzija kromatično višino tona, kot jo definira Meredith[7], tretja dimenzija višino tona, kot jo definira Meredith[8], četrta dimenzija notno trajanje v šestnajstinkah in peta dimenzija glas v večglasju, v katerem se ton pojavi.

Algoritem SIA v večdimenzionalni množici za vsak možen vektor poišče največji vzorec točk, ki se s tem vektorjem preslika v drug vzorec znotraj množice. Množico D najprej uredi in nato iz nje konstruira tabelo vektorjev. Vsaka celica v tabeli vektorjev vsebuje vektor, ki preslika začetni vektor, zapisan v glavi stolpca, v končni vektor, zapisan v glavi vrstice. SIA nato uredi vektorje iz celic tabele in prebere vzorce tako, da za vsak vektor iz urejenega seznama zapiše vse končne točke (vektorje iz glave vrstice), za katere se dani vektor pojavi v celici znotraj tabele.

Algoritem SIATEC za vsak vzorec, ki ga je našel SIA, poišče vse pojavitve tega vzorca. Vse pojavitve dobi tako, da poišče presek vseh stolpcev, ki imajo v glavi zapisano eno od točk iz vzorca.

SIATEC uspešno odkriva vzorce in njihove ponovitve, vendar ne razločuje med glasbeno pomembnimi in nepomembnimi. Več raziskovalcev je nadaljevalo delo s tem algoritmom in poskušalo izmed najdenih vzorcev izluščiti tiste, ki bi jih kot značilne zaznal poslušalec. V nadaljevanju predstavljamo tri take raziskave.

Meredith [37] predstavi dva algoritma, ki izmed vseh vrnjenih vzorcev algoritma SIATEC poiščeta relevantne glasbene motive. COSIATEC v vsaki

[7]Numerična predstavitev tipke na klavirju, ki predstavlja trenutni ton. Ton A0 ima kromatično višino 0. [36]

[8]Numerična predstavitev položaja notne glavice v notnem črtovju. Premik glavice za eno pozicijo navzgor poveča tonsko višino za 1. Ton A0 ima po Meredithu višino 0. Primer: C4 in cis4 imata oba vrednost 23. [36]

iteraciji izmed vrnjenih vzorcev na podlagi kompresije[9], pokritja[10] in kompaktnosti[11] izbere najboljši vzorec, odstrani vse pojavitve vzorca iz množice in ponovno požene SIATEC na preostali množici. Postopek ponavlja, dokler obstajajo relevantni ponavljajoči vzorci. SIATECCOMPRESS požene SIATEC samo enkrat in nato pokriva vzorce od najbolj do najmanj relevantnega, dokler ne doseže želenega pokritja množice. Pri tem se vzorci lahko prekrivajo.

Forth [18] predstavi metodo za razločevanje pomembnih vzorcev izmed vseh, ki jih vrne SIATEC, zasnovano kot reševanje maksimalnega uteženega pokritja množice. SIATEC enkrat požene na vhodnih podatkih in z dobljenimi vzorci pokriva učno skladbo. Pri tem dovoljuje, da je en ton iz skladbe pokrit z več vzorci. Vzorce utežuje glede na glasbeno pomembnost, kjer višja utež vzorca predstavlja večjo pomembnost v strukturi glasbe, in poskuša maksimizirati vsoto uteži pokritja. Pomembnost vsakega vzorca oceni, preden začne pokrivati učno množico. S tem so uteži vseh vzorcev pridobljene v enakem kontekstu. Hevristike za določitev glasbene pomembnosti so zasnovane na merah kompresije in večglasne kompaktnosti[12]. Končna utež vzorca je izračunana kot produkt normaliziranih hevristik. Za hevristike je določena minimalna vrednost, pri kateri se utež še upošteva in s tem izloči premalo zanimive vzorce.

Uporabi požrešni algoritem, ki v vsaki iteraciji izbere množico vzorcev, ki maksimizira razmerje med utežjo pokritja in številom pokritih elementov.

Zadnji korak algoritma je določitev, ali je vzorec primaren ali sekundaren. To je proces, ki grupira podobne vzorce. Prvi izbrani vzorec je vedno prima-

[9]Razmerje med količino podatkov, ki jih potrebujemo, če vse pojavitve vzorca zapišemo kot posamezne točke ali če zapišemo samo osnovni vzorec in za ostale pojavitve podamo samo vektor, s katerim se osnovni vzorec preslika v drugo pojavitev vzorca [38]

[10]Število točk iz množice (skladbe) v vseh pojavitvah vzorca skupaj [38]

[11]Razmerje med številom točk v vzorcu in številom točk na celotni površini, ki jo omejujejo točke vzorca [38]

[12]Kompaktnost, kjer pri polifoni skladbi upoštevamo samo glasove, v katerih se nahaja vzorec.

ren. Za vsak naslednji izbran vzorec je primerjano njegovo skupno pokritje s primarnim vzorcem (točke, ki jih oba pokrivata). Na ta način identificirajo primarni vzorec, ki je novemu najbolj podoben. Če je skupno pokritje večje od arbitrarno določene meje (50 %), potem je novi vzorec razglašen za sekundarni vzorec in grupiran skupaj z najbolj podobnim primarnim vzorcem. Če novo izbrani vzorec ni podoben nobenemu primarnemu vzorcu, je tudi sam razglašen za primarni vzorec. Grupiranje vzorcev predstavlja oceno različnih glasbenih tem v učni skladbi.

Collins idr. [12] v svojem članku poleg že znane pomanjkljivosti SIATECA, vračanja prevelikega števila vzorcev, naslovijo še problem nezaznavanja variacij ritma kot ponovitve istega vzorca. Predstavijo algoritem SIARCT-CFP, ki je nadgradnja algoritma SIATEC.

Za iskanje ponovitev vzorca, ki se od osnovnega razlikujejo v ritmu, uporabijo tehniko simbolnega prstnega odtisa[13] [2]. Izvedejo jo za vsak vzorec, ki ga vrne osnovni algoritem, in dobijo časovne točke, v katerih je verjetnost za začetek pojavitve variacije osnovnega vzorca večja od arbitrarno določene meje.

Za zmanjševanje števila najdenih vzorcev uporabijo grupiranje podobnih vzorcev v en reprezentativen vzorec. Vzorce najprej uredijo po pomembnosti in iz množice neizbranih vzorcev najprej izberejo najbolj pomembnega. S poljubno izbrano mero izračunajo podobnost med izbranim vzorcem in vsemi še neizbranimi vzorci. Vse, ki imajo podobnost večjo od neke meje, grupirajo v ponovitev istega vzorca. Postopek ponavljajo, dokler množica neizbranih vzorcev ni prazna.

Pozneje predstavijo razširitev rešitve na skladbe v avdio zapisu [13], za ka-

[13]Tehnika simbolnega prstnega odtisa je metoda za iskanje ujemajočih vzorcev, ki ne temelji na notnem trajanju in omogoča transpozicijo, časovni zamik, invariantnost na faktor skaliranja in določitev tolerance za interval med časi nastopa dveh tonov glede na originalni vzorec. Njen rezultat je časovna vrsta $S = S_t : t \in T$, kjer je T množica zaporednih točk, ki niso nujno enakomerno porazdeljene. Vrednost S_t označuje verjetnost, da se izbran vzorec začne ob času t. Postopek je podrobno razložen v [2].

tere najprej naredijo transkripcijo in nato nanjo aplicirajo algoritem SIARCT-CFP.

Nieto in Farbood [41, 42] predstavita algoritem, ki išče vzorce v monofoni ali polifoni skladbi, podani v avdio ali simbolnem zapisu. Za skladbo v avdio zapisu algoritem najprej izračuna spektrogram. Nadaljnji postopek je za obe vrsti vhodnih podatkov enak. Bodisi iz simbolnega zapisa bodisi iz spektrograma se izračuna kromagram $C = (c_1, \ldots, c_N)$, ki razdeli vhodni signal oziroma niz na N delov. Iz kromagrama oblikuje transpozicijsko invariantno samopodobnostno matriko S [40]. To je simetrična matrika:

$$S(n, m) = d(c_n, c_m), \forall n, m \in \{1, \ldots, N\}, \tag{2.2}$$

kjer d označuje izbrano razdaljo (Evklidsko, Manhattansko itd.). Dobljena matrika vsebuje diagonalne poti, ki nakazujejo ponavljanja. Nanjo se aplicira diagonalni filter, ki poudari ponavljanja.

Iskanje ponavljanj v matriki poteka s požrešnim iskanjem. Algoritem najprej kreira novo matriko $\hat{S}$, v katero prepiše samo elemente iz S, ki so nad glavno diagonalo. Ostali elementi imajo vrednost 0. Če iščemo vzorce dolžine najmanj ν, je možnih začetkov ustreznih diagonalnih poti v matriki $(N - \nu)(N - \nu - 1)/2$. Za določitev najboljših poti definirajo funkcijo σ, ki je za vsako diagonalno pot izračunana na podlagi elementov podmatrike, katere diagonalo predstavlja ocenjevana pot. Utež, dobljena s funkcijo σ, je normalizirana, da ni odvisna od dolžine ocenjevane poti.

Algoritem obdrži tiste poti, ki imajo utež večjo od meje θ, in jih grupira v pojavitve istega vzorca. Tiste poti, katerih indeksi začetkov in koncev v matriki se ujemajo v drugi dimenziji, predstavljajo ponovitve istega vzorca.

2.4 Hierarhično modeliranje glasbe

Različni vidiki hierarhije v glasbi so obravnavani na številnih znanstvenih področjih: od hierarhične organizacije dela možganov, ki procesira glasbo, in

prepoznave različnih tonskih višin iz frekvenc avdio signala v nevropsihologiji in računski biologiji [49, 35, 26, 3], do dojemanja pomembnosti tonov znotraj lestvice, prepoznavanja akordov, sledenja harmoniji ter strukturiranju glasbe v motive in fraze v muzikologiji in psihologiji [16, 27, 45].

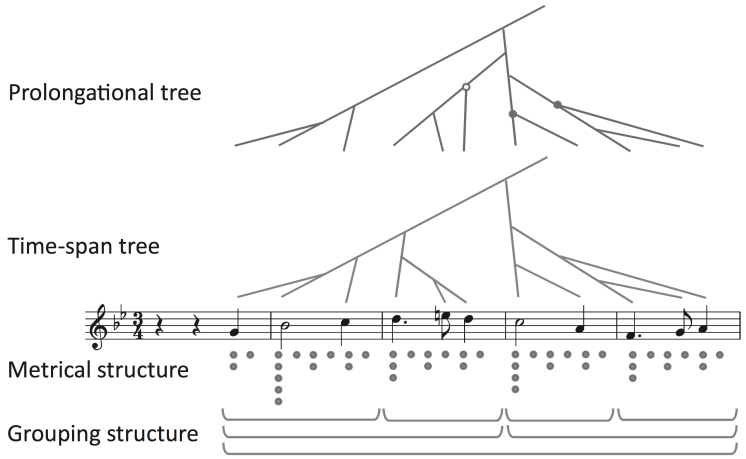
Za nalogo iskanja ponavljajočih vzorcev je relevantno predvsem hierarhično strukturiranje glasbe v motive in fraze, pri katerem posamezne tone grupiramo v vedno večje smiselno zaokrožene celote. Najbolj znani muzikološki metodi, ki strukturo glasbe analizirata s hierarhičnim pristopom in sta pogosto citirani v muzikologiji in psihologiji, sta Shenkerjeva analiza (angl. *Schenkerian analysis*) [46] in Generativna teorija tonalne glasbe (angl. *A generative theory of tonal music*) [31]. Sprožili sta tudi odzive v računalništvu. Za obe obstaja več računalniških implementacij [33, 25, 19], nanju pa se pogosto navezujejo raziskovalci na področju iskanja vzorcev v glasbi.

Schenker v svoji teoriji zajame analizo melodije, kontrapunkta in harmonije. Hierarhijo gradi od najvišjega nivoja navzdol. Začne z osnovno strukturo harmonije I-V-I (tonika-dominanta-tonika)[14]. Nadaljnje nivoje hierarhije oblikuje s transformacijami prejšnjega nivoja, dokler ne dobi celotne skladbe na najnižjem nivoju [30].

Schenker je postopke analize opisal neformalno, zato sta Lerdhall in Jackendorf v Generativni teoriji tonalne glasbe poskušala formalizirati njegovo teorijo, hkrati pa zajeti tudi aspekte, ki jih Schenker ni pokril. Teorija temelji na podzavestnem občutenju skladbe, kot ga doživlja poslušalec. Dimenzije, ki jih v svojem modelu upoštevata, so harmonska struktura (grupiranje v motive na podlagi tonskih višin, angl. *grouping structure*), ritmična struktura (angl. *metrical structure*), kombinacija harmonije in ritma (razvoj harmonije v času – angl. *time-span reduction*) ter napetost (identifikacija delov skladbe, kjer napetost narašča in kjer popušča – angl. *prolongational reduction*). Primer hierarhije za vse štiri dimenzije je prikazan na Sliki 2.4.

Nasprotno od Schenkerja Lerdhall in Jackendorf gradita hierarhijo od spodaj navzgor. Začneta s celotnim glasbenim delom na najnižjem nivoju in

[14]Tonika označuje prvi ton v izbrani tonaliteti, dominanta pa peti ton.



Slika 2.4: Primer hierarhije glasbenih struktur po Generativni teoriji tonalne glasbe [19].

s pomočjo odločitvenih pravil modelirata vse štiri dimenzije glasbe, da dobita razdelitev na glasbene strukture na najvišjem nivoju [30].

Poglavje 3

Kompozicionalni hierarhični model

Kompozicionalni hierarhični model (CHM – angl. *Compositional Hierarchical Model*) je globoka arhitektura, namenjena pridobivanju informacij iz glasbe. Zasnovan je na podlagi hierarhičnega modela s področja računalniškega vida [17], prilagoditev za področje pridobivanja informacij iz glasbe pa so predstavili Pesek idr. [43]. Predstavljeni model je namenjen vhodnim podatkom, podanim v avdio zapisu. Do sedaj je bil uspešno uporabljen za prepoznavanje akordov in osnovnih frekvenc. Njegova prednost pred ostalimi globokimi arhitekturami je v transparentnosti in relativnosti pridobljenega znanja. Transparentnost omogoča lažje razumevanje modela in interpretacijo rezultatov ter preverjanje njihove pravilnosti. Relativnost omogoča učenje koncepta namesto natančnih vrednosti. Na ta način se zmanjša prostorska zahtevnost modela, hkrati pa lahko z enim konceptom pokrije več različnih delov učne množice. V magistrski nalogi smo obstoječi model prilagodili za branje in procesiranje dvodimenzionalnih podatkov, podanih v simbolnem glasbenem zapisu, ter ga uporabili za reševanje naloge iskanja vzorcev v glasbi. Pri tem smo ohranili osnovno strukturo in metode, predstavljene v [43].

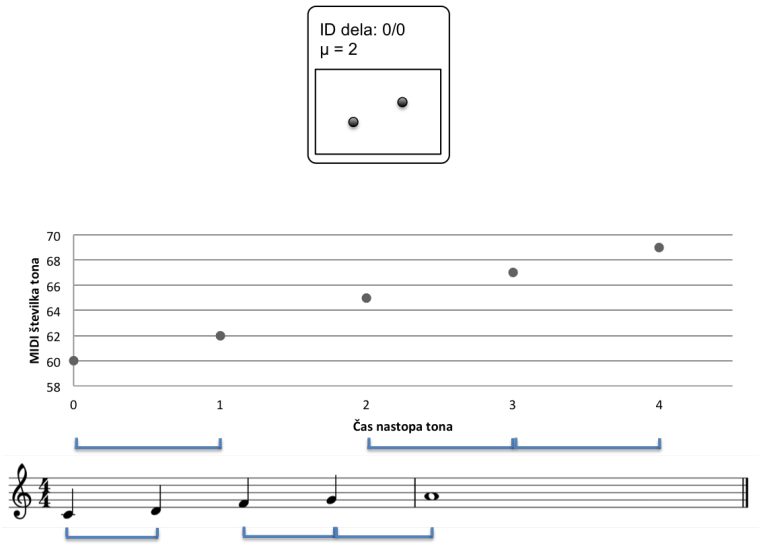
3.1 Kompozicionalni hierarhični model za simbolni glasbeni zapis

Kompozicionalni hierarhični model za simbolni glasbeni zapis (SymCHM) hierarhično predstavi glasbo, podano v simbolnem glasbenem zapisu, od posameznih gradnikov (posamezen ton) na spodnjem nivoju do kompleksih struktur (glasbenih tem in motivov) na višjih nivojih. V fazi učenja se model nenadzorovano nauči vzorce (relativne odvisnosti med toni v vzorcu), ki se večkrat ponovijo v vhodnih podatkih. Učenje lahko zaženemo na eni ali več skladbah naenkrat, odvisno od želenega obsega iskanja ponavljajočih vzorcev. Na zgrajenem modelu nato izvedemo inferenco na učnih podatkih, da dobimo absolutne lokacije vzorcev v učni množici, ali pa na drugi testni množici, da vidimo, če se naučeni vzorci pojavijo tudi v drugi množici in pridobimo njihove absolutne lokacije. V primeru iskanja ponavljajočih vzorcev znotraj ene skladbe, ki se mu posvečamo v magistrski nalogi, učenje zaženemo za vsako skladbo posebej in nato na isti skladbi izvedemo še inferenco.

3.1.1 Struktura modela

Model temelji na predpostavki, da lahko glasbo razbijemo na več osnovnih gradnikov, ki jih imenujemo *deli*. Del lahko opisuje posamezen gradnik glasbe (ton) ali kompleksno kompozicijo gradnikov (vzorec, zgrajen iz več tonov). Del predstavlja nek koncept, ki se pojavi v skladbi, ne pa natančnih vrednosti tonov oziroma mest v skladbi. V primeru iskanja vzorcev so to razmerja med višinami tonov, kot je prikazano na Sliki 3.1.

Glede na kompleksnost lahko dele razvrstimo v več nivojev, od najmanj do najbolj kompleksnih. Deli na višjih nivojih so sestavljeni kot kompozicija delov z nižjega nivoja. Model sestoji iz vhodnega nivoja $\mathcal{L}_0$ in več sestavljenih nivojev $\{\mathcal{L}_1, \ldots, \mathcal{L}_N\}$.



Slika 3.1: Od spodaj navzgor je na sliki prikazan notni zapis, nad njim grafična predstavitev tega zapisa kot MIDI višina tona v času, na vrhu pa *del*, kot ga definiramo v kompozicionalnem hierarhičnem modelu za simbolni glasbeni zapis. Del vsebuje identifikacijsko številko dela, odmik med poddeloma μ, merjen v številu poltonov, in grafično predstavitev odmikov med poddeli. Primer dela na sliki predstavlja dva tona (noti), ki se pojavita z razmikom dveh poltonov (zato je $\mu = 2$). Toni se v razmiku dveh poltonov v primeru na sliki pojavijo trikrat – pri časih 0, 2 in 3.

Sestavljeni nivoji

Vsak sestavljen nivo $\mathcal{L}_n$ vsebuje množico delov $\{P_1^n, \ldots, P_M^n\}$, kjer je vsak del P_i^n sestavljen iz več *poddelov* z nivoja $\mathcal{L}_{n-1}$, hkrati pa lahko nastopa kot poddel v poljubnem številu delov na nivoju $\mathcal{L}_{n+1}$. Del P_i^n definiramo kot

$$P_i^n = \{P_{k_0}^{n-1}, \{P_{k_j}^{n-1}, (\mu_j, \sigma_j)\}_{j=1}^{K-1}\}. \tag{3.1}$$

P_i^n je sestavljen iz K delov z nivoja $\mathcal{L}_{n-1}$, ki jih imenujemo poddeli dela P_i^n. Kompozicije poddelov so definirane kot relativne razdalje (odmiki) med poddelom $P_{k_0}^{n-1}$ in poddeli $P_{k_1}^{n-1}, \ldots, P_{k_K\ 1}^{n-1}$. Poddel $P_{k_0}^{n-1}$ imenujemo *centralni del* kompozicije. Odmiki glede na centralni del so določeni s parametri $\mu_1, \ldots, \mu_{K-1}$, ki predstavljajo absolutni odmik od centralnega dela, in $\sigma_1, \ldots, \sigma_{K-1}$, ki modelirajo dovoljeno odstopanje od μ_i. V okviru magistrske naloge smo implementirali konstantno vrednost $\sigma_i = 0, \forall i$, torej ne dovoljujemo odstopanja od vrednosti μ_i. Vse kompozicije in pripadajoči parametri so naučeni z nenadzorovanim učenjem. Primer enostavne kompozicije je prikazan na Sliki 3.2.

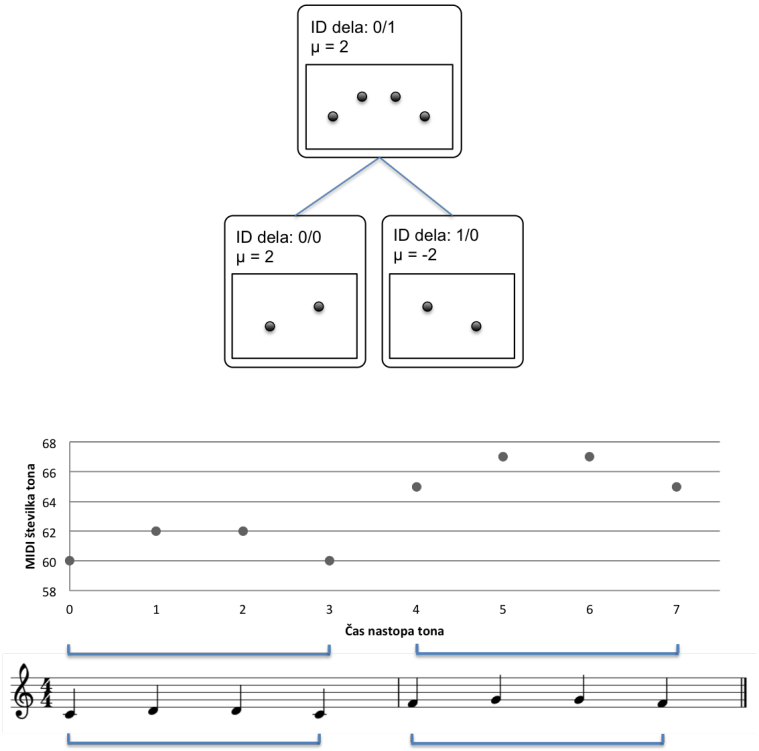
Del se *aktivira*, ko je koncept (vzorec), ki ga predstavlja, najden v vhodnih podatkih. Aktivacija ima tri komponente: *lokacijo* (višino tona, angl. *pitch*) in *čas* (čas nastopa tona, angl. *onset*), ki preslikata del (vzorec) v konkretno pojavitev vzorca v vhodnih podatkih, ter moč aktivacije. Primer kompozicije s pripadajočimi aktivacijami je predstavljen na Sliki 3.3.

Del se lahko aktivira samo, če se aktivirajo vsi njegovi poddeli z močjo, večjo od 0 (ta pogoj lahko omilimo, kot je opisano v podpoglavju 3.1.3). Del ima lahko več aktivacij na različnih lokacijah, kar označuje več ponovitev vzorca v vhodnih podatkih.

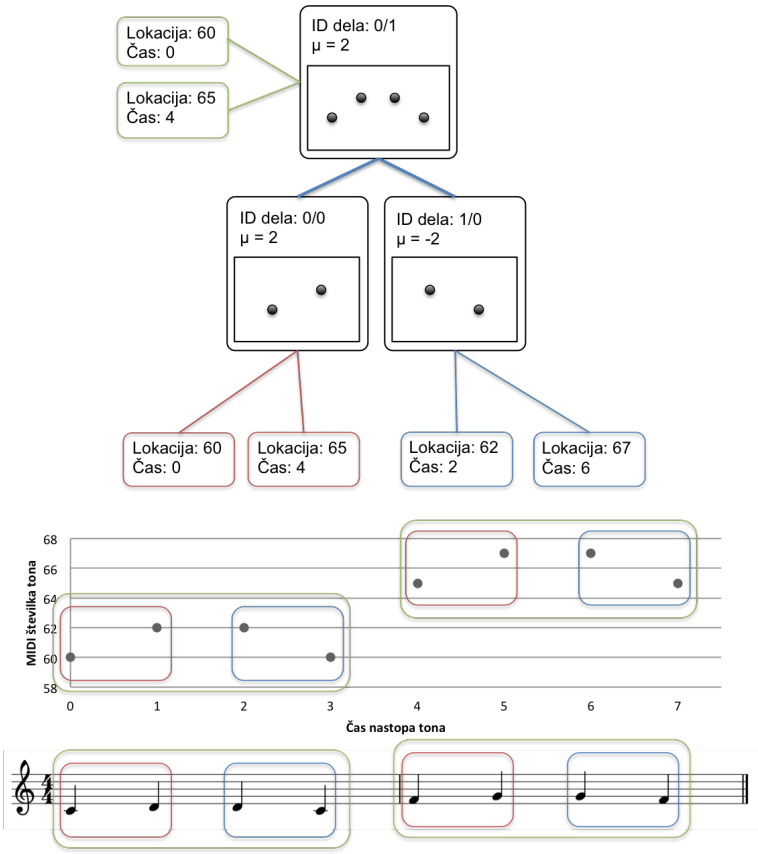
Lokacija aktivacije $A_L(P_i^n)$ in čas aktivacije $A_T(P_i^n)$ dela P_i^n sta definirani kot

$$\begin{aligned} A_L(P_i^n) &= A_L(P_{k_0}^{n-1}), \\ A_T(P_i^n) &= A_T(P_{k_0}^{n-1}), \end{aligned} \tag{3.2}$$

lokacija in čas dela sta torej enaka lokaciji in času centralnega poddela. Na ta način centralni deli kompozicije propagirajo svoje lokacije in čase navzgor



Slika 3.2: Slika prikazuje primer enostavne kompozicije. Del 0/1 je sestavljen iz poddelov 0/0 in 1/0 z odmikom $\mu = 2$ (prvi ton dela 1/0 se v kompoziciji pojavi 2 poltona višje od prvega tona dela 0/0).



Slika 3.3: Primer aktivacij dela in njegovih poddelov. Del opisuje relativne odmike med toni, aktivacija pa vzorec, ki ga predstavlja del, preslika v absolutne vrednosti (konkretne pojavitve vzorca v skladbi). Aktivacije so predstavljene kot barvni pravokotniki, povezani s pripadajočim delom. Vsaka vsebuje informacijo o lokaciji in času, kjer se del aktivira. Barvni okvirji v notnem črtovju in pripadajočem grafu prikazujejo aktivacije, ki so v hierarhiji obrobljene z enako barvo.

po hierarhiji. Propagiranje lokacij skozi centralne dele predstavlja indeksni mehanizem, ki omogoča učinkovito analizo aktivacij od vrha hierarhije navzdol.

Moč aktivacije je definirana kot utežena vsota moči aktivacij poddelov:

$$A_M(P_i^n) = tanh\left(\frac{1}{K}\sum_{i=0}^{K-1} w_i A_M(P_{k_i}^{n-1})\right), \tag{3.3}$$

kjer so uteži w_i definirane z ujemanjem lokacij poddelov δ_{L_i} s kompozicijskimi parametri μ in σ:

$$\begin{aligned}
\delta_{L_i} &= A_L(P_{k_i}^{n-1}) – A_L(P_{k_0}^{n-1}), \\
\delta_{T_i} &= A_T(P_{k_i}^{n-1}) – A_T(P_{k_0}^{n-1}), \\
w_i &= \begin{cases} 1: & i = 0 \\ \mathcal{N}(\delta_{L_i}, \mu_i, \sigma_i): & i > 0 \wedge \delta_{T_i} < \tau_W \\ 0: & \text{sicer} \end{cases}
\end{aligned} \tag{3.4}$$

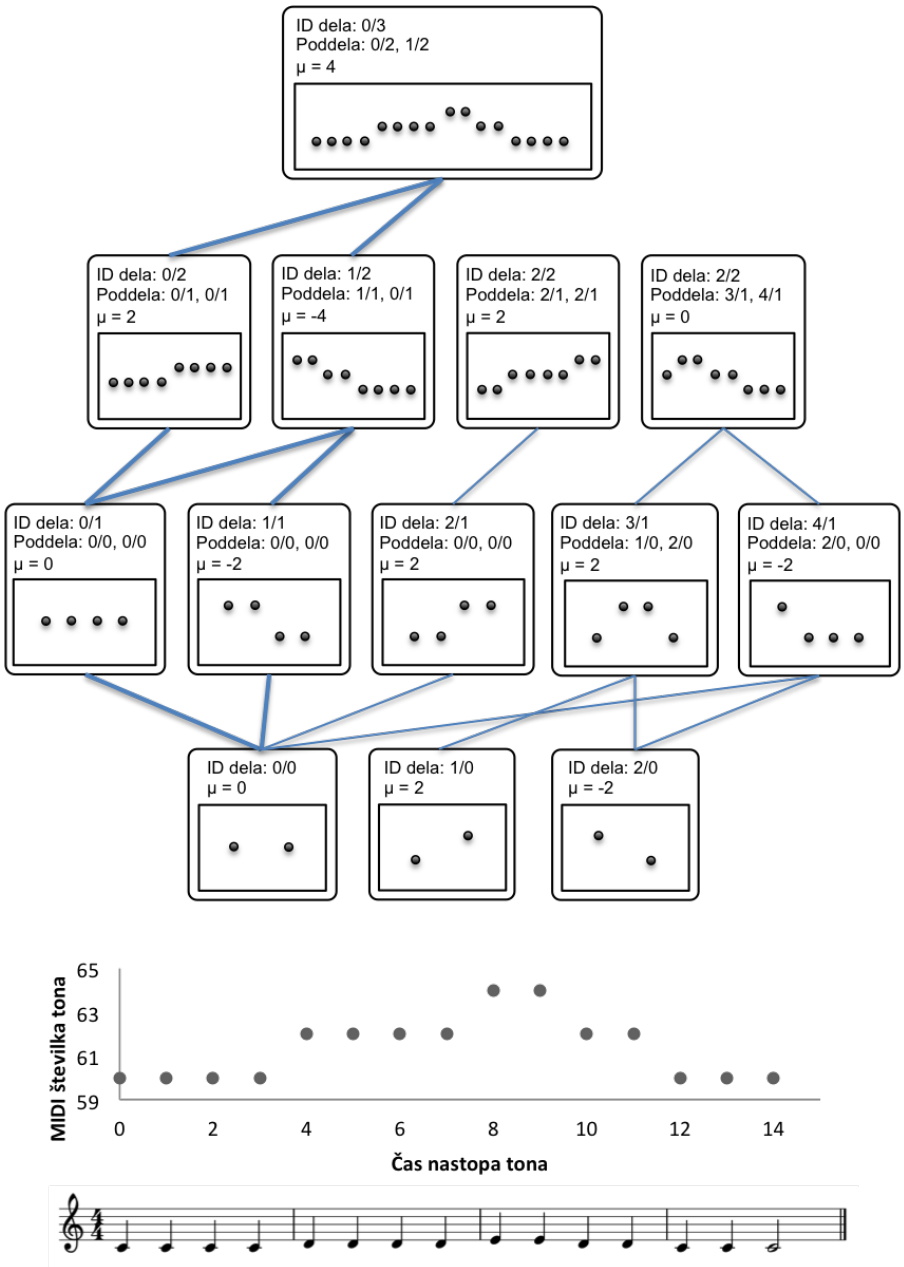
Funkcija $\mathcal{N}$ je nenormalizirana Gaussova funkcija, ki določi manjšo utež delom, ki odstopajo od naučenih parametrov kompozicije μ_i in σ_i. Parameter τ_W predstavlja časovno okno, s katerim omejimo časovno oddaljenost opazovanih aktivacij.

Vhodni nivo

Vhodni nivo L_0 modelira glasbo, podano v simbolnem zapisu, kot zaporedje dogodkov (tonov) $N \in \mathcal{S}$, definiranih z višino tona N_o in časom nastopa tona N_p. Vhodni nivo sestoji iz enega samega dela P_1^0, ki se aktivira na vseh dogodkih iz vhodne reprezentacije:

$$\forall N \in (S) = \begin{cases} A_L(P_1^0) = N_p \\ A_T(P_1^0) = N_o \\ A_M(P_1^0) = 1. \end{cases} \tag{3.5}$$

Pri tem je lokacija aktivacije enaka višini tona, njen čas pa času nastopa tona. Moč vseh aktivacij na vhodnem nivoju je enaka 1, vendar se jo lahko



Slika 3.4: Primer hierarhije, ki je naučena na skladbi Kuža pazi. Model bi v praksi na višjih nivojih zgeneriral več delov, vendar jih je zaradi preglednosti prikazanih le nekaj.

prilagodi, da predstavlja dinamiko skladbe, tako da večjo vrednost pripišemo poudarjenim tonom.

Primer zgrajene hierarhije, s katero se kompozicionalni hierarhični model za simbolni glasbeni zapis nauči celotno melodijo skladbe Kuža pazi, je prikazan na Sliki 3.4.

3.1.2 Učenje

Model je zgrajen po nivojih od spodaj navzgor z nenadzorovanim učenjem iz enega ali več simbolnih zapisov, začenši z nivojem $\mathcal{L}_1$. Učni proces je problem optimizacije, kjer želimo za vsak nivo poiskati nabor kompozicij, ki pokrijejo največjo količino informacij z vhodnega nivoja. Učenje vsakega nivoja sestoji iz dveh delov: iskanja množice kandidatov za kompozicije in izbora kompozicij, ki pokrijejo največjo količino informacij iz vhodnih podatkov.

Iskanje kandidatov za kompozicije

Iskanje nabora kandidatov za kompozicije je implementirano ločeno za prvi nivo in za višje nivoje.

Kandidati za nivo $\mathcal{L}_1$ so generirani iz kombinacij tonov iz skladbe, ki se dovolj pogosto pojavijo znotraj izbranega časovnega okna τ_W. Za vsako tonsko višino kreiramo histogram sopojavitev vseh tonskih višin znotraj izbranega časovnega okna in kreiramo kandidate iz tistih višin, katerih delež

je večji od izbrane meje τ_L. Postopek je prikazan v Algoritmu 1.

MidiObseg ← 128; `// Število vseh možnih tonskih višin.`
počisti seznam kandidatov;
inicializiraj histograme; `// Kreiraj prazen seznam dolžine MidiObseg, v katerem je vsak element zopet prazen seznam dolžine MidiObseg`
foreach *skladba* ∈ *učna množica* **do**
  foreach *ton* ∈ *skladba (urejeno po času)* **do**
    `// ton ima lastnosti "višina"in "čas"`
    $t_A \leftarrow$ trenutni ton;
    $t_B \leftarrow t_A$;
    while *(v skladbi* $\exists$ *ton, ki sledi* t_B*)* $\wedge$ *(*t_B*.čas –* t_A*.čas)* $\leq \tau_W$ **do**
      $t_B \leftarrow$ ton, ki sledi t_B;
      histogram[t_A.višina][t_B.višina] ←
      histogram[t_A.višina][t_B.višina] + 1;
    end
  end
end
for n_A *od 0 do (MidiObseg-1)* **do**
  $hist_A \leftarrow$ histogram[n_A];
  skaliraj $hist_A$ na interval od 0 do 1;
  $max_A \leftarrow$ maksimalna vrednost v $hist_A$;
  while $max_A > \tau_L$ **do**
    izračunaj μ;
    if $\nexists$ *kandidat z enakim* μ **then**
      kreiraj kandidata;
    end
    maksimalna vrednost v $hist_A \leftarrow 0$; `// samo za upoštevano pojavitev, ne za vse pojavitve`
    $max_A \leftarrow$ maksimalna vrednost v $hist_A$;
  end
end

Algorithm 1: Generiranje kandidatov za prvi nivo. Parameter τ_W predstavlja časovno okno, v katerem opazujemo sopojavitve, τ_L pa mejo deleža sopojavitev tonskih višin za kreiranje novega kandidata.

S časovnim oknom τ_W omejimo kompleksnost iskanja. Med posameznimi toni glasbeno relevantnega vzorca običajno ni velikih časovnih intervalov, zato smatramo, da z omejitvijo iskanja na okno ne izpustimo relevantnih vzorcev. Okno je definirano po nivojih in se eksponentno povečuje, saj se s premikanjem po hierarhiji navzgor vzorci daljšajo.

Kandidati na nivoju $\mathcal{L}_n$ se zgenerirajo po podobnem postopku kot na prvem nivoju, le da tokrat namesto kombinacij tonov iz skladbe opazujemo kombinacije parov aktivacij z nivoja $\mathcal{L}_{n-1}$. Na učni množici najprej izvedemo inferenco[1] do nivoja $\mathcal{L}_{n-1}$ in opazujemo sopojavitve aktivacij delov z nivoja $\mathcal{L}_{n-1}$ znotraj časovnega okna τ_W skozi celotno skladbo. Izračunamo histograme sopojavitev aktivacij glede na razdalje med lokacijami aktivacij. Nove kompozicije tvorimo iz kombinacij delov z nivoja $\mathcal{L}_{n-1}$, za katere delež sopojavitev aktivacij preseže mero τ_L. V splošnem ima en del lahko poljubno veliko poddelov, v okviru magistrske naloge pa smo omejili, da je vsak del sestavljen iz natanko dveh poddelov. Kompozicijski parametri μ in σ so pridobljeni iz pripadajočih histogramov, nove kompozicije pa so dodane v množico kandidatov za kompozicije $\mathcal{C}$.

Izbor kompozicij

S požrešnim pristopom iz množice $\mathcal{C}$ izberemo manjšo podmnožico kompozicij, ki pusti minimalno količino informacij z vhodnega nivoja nepokrito. V vsaki iteraciji izberemo kompozicijo iz množice $\mathcal{C}$, ki največ prispeva k pokritju učne množice, in jo dodamo v nov nivo. Kompozicije dodajamo, dokler ne pokrijemo zadostnega deleža učne množice oziroma dokler doprinos novega kandidata ne pade pod določeno mejo. Učni proces napreduje po nivojih navzgor, dokler ni doseženo želeno število nivojev. Postopek izbora

[1]Proces računanja aktivacij delov iz vhodnih podatkov. Postopek je razložen v razdelku 3.1.3.

kandidatov je predstavljen v Algoritmu 2.

izbraniKandidati $\leftarrow \emptyset$;
preostaliKandidati $\leftarrow$ vsi kandidati;
prejsnjePokritje $\leftarrow 0$;
trenutnoPokritje $\leftarrow 0$;
while $(trenutnoPokritje < \tau_1) \wedge$
$(|trenutnoPokritje - prejsnjePokritje| \geq \tau_2) \wedge$
$(|preostaliKandidati|) > 0$ **do**
  prejsnjePokritje $\leftarrow$ trenutnoPokritje;
  pokritja[preostaliKandidati] $\leftarrow$ izracunajPokritja; `// Za vsakega preostalega kandidata izračunaj, kolikšen delež vhodnih podatkov pokrijejo njegove aktivacije skupaj z aktivacijami že izbranih kandidatov`
  urediPokritja; `// Uredi pokritja v padajočem vrstnem redu`
  trenutnoPokritje $\leftarrow$ pokritja[najboljšiKandidat];
  if $(|trenutnoPokritje - prejsnjePokritje| \geq \tau_2)$ **then**
    izbraniKandidati.Dodaj(najboljšiKandidat);
  end
  preostaliKandidati.Odstrani(najboljšiKandidat);
end

Algorithm 2: Izbor kandidatov. Parameter τ_1 predstavlja delež vhodnih podatkov, ki jih želimo pokriti z izbranimi kandidati, τ_2 pa najmanjši doprinos kandidata, ki ga še upoštevamo.

3.1.3 Inferenca

Inferenca je proces računanja aktivacij delov iz vhodnih podatkov glede na enačbi 3.2 in 3.3. Za vsak del preslika koncept, ki ga del predstavlja, v absolutne lokacije v vhodnih podatkih ter izračuna moč aktivacij. Obdrži tiste aktivacije, ki imajo moč večjo od meje, ki jo postavimo. Računanje poteka po nivojih od spodaj navzgor, kjer simbolni vhodni podatki (višina tona in čas nastopa tona) služijo kot podatki za nivo $\mathcal{L}_0$. Izračunane lokacije, časi in moči aktivacij nudijo vpogled v vzorce, ki ga aktivirani deli predstavljajo,

ali pa so uporabljeni za nadaljnje procesiranje.

Inferenca je izpostavljena kot samostojen proces z namenom, da je za model, naučen na eni skladbi, možno izvesti inferenco na drugi skladbi za nekatere druge naloge, kot je na primer analiza vzorcev preko več različnih skladb, iskanje plagiatov in prepoznava skladatelja.

Tako kot generiranje hierarhije je tudi inferenca implementirana ločeno za prvi nivo in za višje nivoje. Na prvem nivoju poiščemo aktivacije tako, da za vsak ton skladbe pogledamo vse tone, ki mu sledijo in so od njega odmaknjeni za manj kot τ_W. Interval med višinama obeh tonov primerjamo z deli na nivoju $\mathcal{L}_1$. Če najdemo del, katerega odmik μ je enak intervalu med višinama tonov, smo našli aktivacijo tega dela.

Na nivoju $\mathcal{L}_n$ nove aktivacije kreiramo kot kombinacije aktivacij z nivoja $\mathcal{L}_{n-1}$. Za vsak del pogledamo kombinacije vseh aktivacij njegovih poddelov s prejšnjega nivoja in iz njih kreiramo nove začasne aktivacije. Če je moč novo nastalih aktivacij večja od izbrane meje, jih dodamo v nivo, sicer jih izpustimo.

Med inferenco se lahko izvedeta dva dodatna mehanizma, ki povečata zmožnost napovedovanja in robustnost modela: halucinacija in inhibicija.

Halucinacija

Privzeto obnašanje modela pri izračunu aktivacij je zelo striktno - del se aktivira samo, če se vsi njegovi poddeli aktivirajo z močjo, večjo od 0. Halucinacija omehča ta pogoj in omogoči kreiranje aktivacije tudi v primeru, ko niso aktivirani vsi poddeli. S tem omogoči iskanje variacij posameznega vzorca. Model generira aktivacije delov tako, da se kar najbolj prilegajo podatkom z vhodnega nivoja, kjer so fragmenti, ki niso prisotni v aktivacijah poddelov, halucinirani.

Za uskladitev strukture modela z vhodnimi podatki izračunamo *pokritje* aktivacij enega dela kot množico lokacij in časov aktivacij nivoja $\mathcal{L}_0$, ki so povzročile aktiviranje dela. To množico dobimo z rekurzivnim sledenjem drevesu aktiviranih poddelov po hierarhiji navzdol do nivoja $\mathcal{L}_0$ po indeksni

strukturi, zapisani v lokacijah centralnih delov:

$$\begin{aligned} A_C(P_i^n) &= \bigcup_{j=0}^{K-1} A_C(P_{k_j}^{n-1}) \\ A_C(P_1^0) &= \{(L,T) : \exists A_L(P_1^0) \wedge \exists A_T(P_1^0)\}. \end{aligned} \tag{3.6}$$

Halucinacija sprosti pogoje, pod katerimi se del lahko aktivira. Nadzira jo parameter τ_H, ki je lahko definiran ločeno za posamezen nivo in se med inferenco lahko posodablja. S halucinacijo se del P_i^n lahko aktivira, ko delež dogodkov z nivoja $\mathcal{L}_0$, ki jih pokriva P_i^n, preseže τ_H:

$$\frac{\left|\{k : k \in A_C(P_i^n) \wedge (X(k) > 0)\}\right|}{\left|A_C(P_i^n)\right|} \geq \tau_H. \tag{3.7}$$

Če postavimo τ_H na 1, dobimo privzeto obnašanje modela (vsi pokriti dogodki morajo biti prisotni v v vhodnih podatkih, da se del aktivira). Zmanjšanje vrednosti parametra se odraža v povečanju števila haluciniranih aktivacij in s tem večjim variacijam vzorcev.

Inhibicija

Inhibicija omogoča prečiščevanje hipotez z zmanjševanjem števila redundantnih aktivacij delov na posameznih nivojih. Čeprav algoritem penalizira dele, ki redundantno pokrivajo vhodne podatke, se še vedno pojavljajo nekateri redundantni deli. Posledično lahko med inferenco posamezen nivo kreira številne redundantne aktivacije, ki pokrivajo iste dogodke z vhodnega nivoja.

Aktivacija dela P_i^n je inhibirana, ko drug del P_j^n z istega nivoja pokriva iste dogodke iz vhodnih podatkov, ampak z večjo močjo aktivacije.

$$\exists\{P_j^n..P_k^n\} : \wedge \begin{cases} \frac{\left|A_C(P_i^n) \backslash \{A_C(P_j^n)..A_C(P_k^n)\}\right|}{\left|A_C(P_i^n)\right|} < \tau_I \\ A_M(P_{j..k}^n) > A_M(P_i^n) \end{cases}, \tag{3.8}$$

Pri tem parameter τ_I uravnava stopnjo inhibicije. Na primer vrednost parametra $\tau_I = 0,5$ povzroči, da se aktivacija inhibira, če je polovica podatkov z vhodnega nivoja, ki jih pokriva ta aktivacija, že pokrita z drugo močnejšo aktivacijo.

3.1.4 Izbor vzorcev

Po generiranju hierarhije imamo v delih zapisana relativna razmerja med višinami tonov, ki predstavljajo vzorce iz skladbe. Pri inferenci za vsak del poiščemo aktivacije, ki predstavljajo absolutno lokacijo vzorca – to so pojavitve posameznega vzorca. Pred izpisom rezultata filtriramo vzorce, da izberemo samo tiste, ki jih smatramo kot glasbeno relevantne. Za nalogo iskanja ponavljajočih tem in vzorcev nas zanimajo vzorci, ki se v skladbi ponovijo večkrat, zato kot relevantne vzamemo tiste dele, ki imajo več kot tri aktivacije.

Vzorcev, ki so zelo kratki (na primer sestavljeni iz dveh tonov) in se večkrat ponovijo, je v skladbi veliko, vendar jih poslušalec ne zazna kot pomembne. Kot vzorce, ki bi jih poslušalec zaznal, smatramo daljše vzorce z nivojev od vključno $\mathcal{L}_4$ do vključno $\mathcal{L}_6$. Deli, ki nastopajo kot poddeli v delih na višjem nivoju, opisujejo isti vzorec, le da so krajši, zato jih pri izpisu rezultata izpustimo. Iz delov z nivojev $\mathcal{L}_4$, $\mathcal{L}_5$ in $\mathcal{L}_6$ tvorimo unijo in nato odstranimo dele, ki se pojavijo kot poddeli na višjih nivojih: z nivoja $\mathcal{L}_5$ odstranimo dele, ki tvorijo kompozicije na nivoju $\mathcal{L}_6$, nato pa še dele z nivoja $\mathcal{L}_4$, ki tvorijo kompozicije na nivoju $\mathcal{L}_5$. Izbrane kompozicije tvorijo unijo:

$$\bigcup_{l \in \{4,5,6\}} \{P_i^l : (P_i^l \in \mathcal{L}_l) \wedge (P_i^l \notin P_j^{l+1})\} \tag{3.9}$$

Algoritem iz unije izbranih kompozicij generira seznam ponavljajočih vzorcev, kjer vsak del predstavlja vzorec, njegove aktivacije pa posamezne ponovitve vzorca.

3.2 Implementacija modela

Model je napisan v programskem jeziku C#. Implementirali smo konzolno verzijo in verzijo z uporabniškim vmesnikom. Uporaba uporabniškega vmesnika je podrobneje opisana v poglavju 3.2.1. Pri zagonu v konzoli lahko podamo tri parametre: pot do direktorija z vhodnimi podatki, pot do di-

rektorija za datoteke z rezultati ter opcijsko pot do datoteke s parametri za učenje. Če tretjega parametra ne podamo, se pri učenju uporabijo privzete vrednosti parametrov.

Vhodni podatki so podani v csv datoteki s petimi stolpci: čas nastopa tona (angl. *ontime*), merjen v številu dob od začetka skladbe, MIDI številka tona, višina tona, kot jo definira Meredith[2], trajanje tona, merjeno v številu dob in zaporedna številka takta. Vsaka csv datoteka predstavlja eno skladbo.

Rezultat algoritma se zapiše v tekstovno datoteko. Najprej se zapišejo vse ponovitve prvega vzorca, nato vse ponovitve drugega itd. Za vsako ponovitev se zapišejo vsi pripadajoči toni kot pari časov nastopa tona in tonske višine. Primer oblike izhodne datoteke je v Prilogi A.

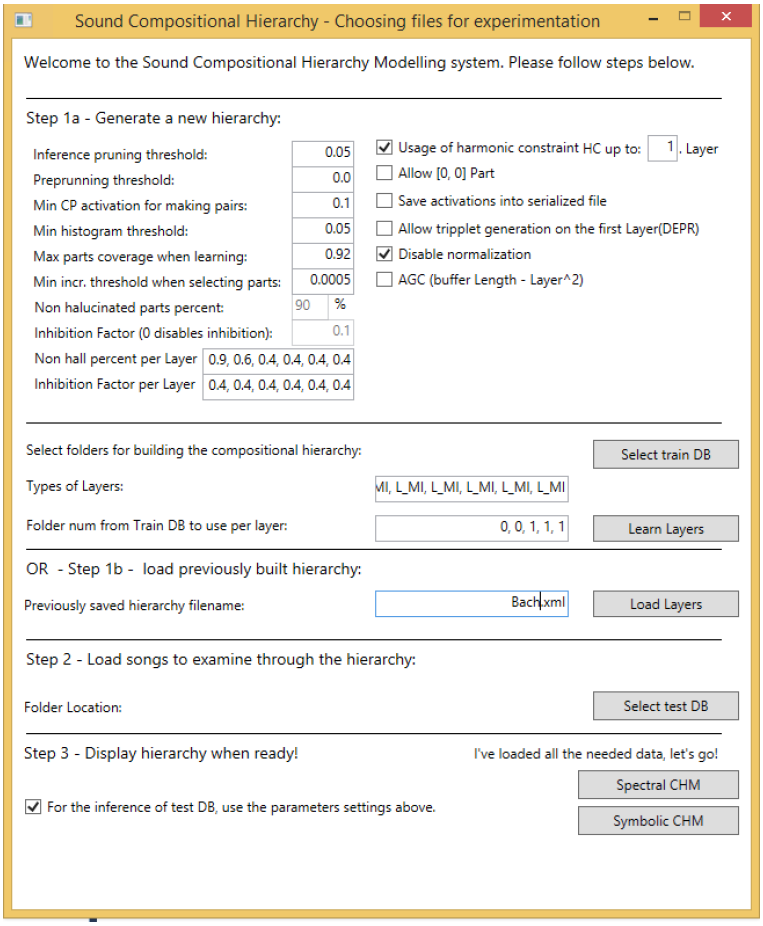
3.2.1 Vizualizacija

V okviru magistrske naloge smo z namenom lažjega pregledovanja vmesnih in končnih rezultatov modela ter odpravljanja napak naredili tudi vizualizacijo modela. Implementirana je v programskem jeziku C# s podporo WPF (Windows Presentation Foundation), za lažjo manipulacijo grafov pa smo uporabili knjižnico GraphSharp.

Ob zagonu vizualizacije se odpre začetno okno, prikazano na Sliki 3.5, ki nas vodi skozi tri korake, potrebne za prikaz vizualizacije. V prvem koraku izvedemo učenje nove hierarhije ali naložimo shranjeno hierarhijo. Če izberemo učenje nove hierahije (korak 1a – angl. *Step 1a*), izberemo vrednosti parametrov za učenje v prvem razdelku, nato pa v drugem razdelku vrste in število nivojev ter učno množico in zaženemo učenje. Če želimo naložiti obstoječo hierarhijo, v koraku 1b izberemo ime datoteke, v katero smo shranili hierarhijo, in jo naložimo.V drugem koraku izberemo testno množico (množico, na kateri izvedemo inferenco), v tretjem koraku pa prikažemo vizualizacijo kompozicionalnega hierarhičnega modela za simbolni glasbeni zapis s klikom na gumb “Symbolic CHM”.

Odpre se novo okno, prikazano na Sliki 3.6. V osrednjem delu okna se

[2]Razložena v poglavju 2.3



Slika 3.5: Začetno okno vizualizacije modela.

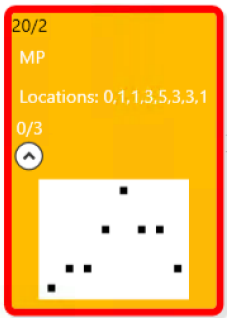
Slika 3.6: Glavno okno vizualizacije naučene hierarhije. V osrednjem delu okna se izriše graf, ki prikazuje naučeno hierarhijo. Pod njim je izris celotne skladbe, moder okvirček na dnu pa je prostor za izpis lokacij in časov aktivacij.

izriše graf, ki prikazuje naučeno hierarhijo. Pod njim je izris celotne skladbe, modri okvirček na dnu pa je rezerviran prostor za izpis lokacij in časov aktivacij.

3.2.2 Izris hierarhije

Deli so prikazani kot vozlišča grafa z vpisanimi osnovnimi podatki o delu: identifikacijska številka dela, vrsta dela, relativne lokacije, zaporedna prikazana aktivacija od vseh aktivacij tega dela (več v razdelku 3.2.4) ter grafični prikaz odmikov med poddeli, ki ga odpremo ali skrijemo s klikom na puščico. Barva dela po barvni shemi "jet"označuje moč aktivacije. Primer dela je prikazan na Sliki 3.7.

Deli so organizirani v nivoje, od najnižjega na dnu grafa do najvišjega nivoja na vrhu grafa. Puščice med nivoji označujejo, kateri poddeli sestavljajo del na višjem nivoju. Približan izsek iz hierarhije je prikazan na Sliki 3.8. Če



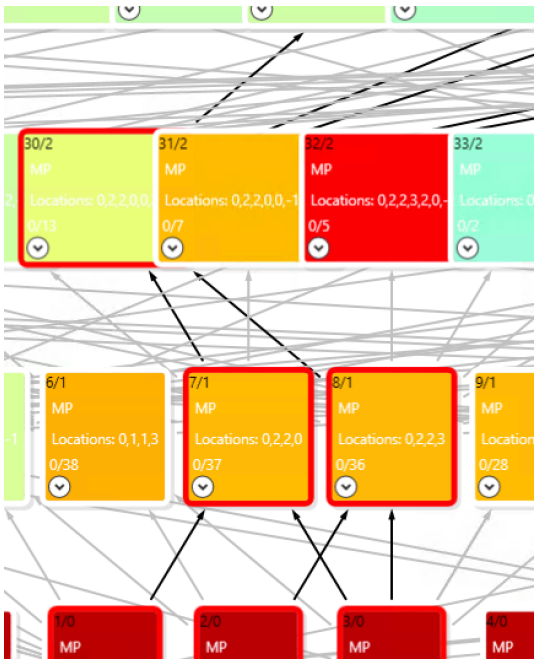
Slika 3.7: Izris vozlišča grafa, ki predstavlja del. Vsebuje informacije o identifikacijski številka dela, vrsti dela, relativnih lokacijah, zaporedni prikazani aktivaciji od vseh aktivacij tega dela ter grafični prikaz odmikov med poddeli.

se z miško postavimo na enega od delov, se odebelijo povezave do poddelov na nižjih nivojih, ki sestavljajo ta del, ter povezave do delov na višjih nivojih, v katerih ta del nastopa kot poddel. Del, na katerem stojimo z miško, in njegovi poddeli na nižjih nivojih dobijo tudi rdečo obrobo.

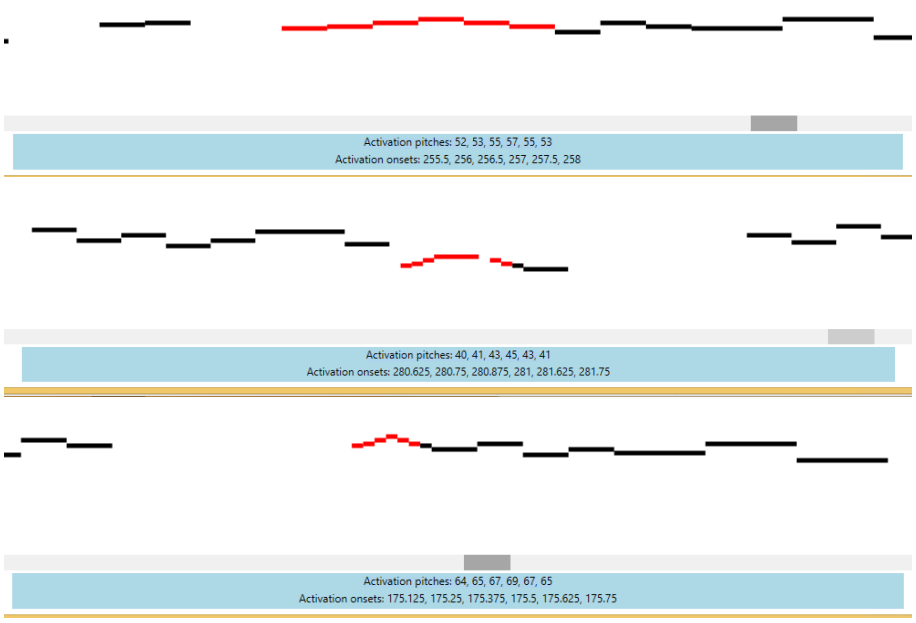
Ob prvem prikazu vizualizacije je hierarhija skalirana tako, da nivo z največjim številom poddelov zavzame celotno širino okna. Z drsnikom v levem zgornjem kotu (vidnem na Sliki 3.6) lahko graf hierarhije povečujemo ali zmanjšujemo. S klikom v prazno polje ali na posamezni del in potegom lahko premikamo celotni graf ali posamezne dele.

3.2.3 Izris skladbe

Izris skladbe je viden na Sliki 3.6 pod izrisom grafa hierarhije. Za vsak ton se izriše črn pravokotnik, kjer koordinata y označuje vrednost višine tona (predstavljena kot MIDI številska vrednost), koordinata x čas nastopa tona, dolžina pravokotnika pa dolžino tona v šestnajstinkah. Z drsnikom na dnu okna se lahko premikamo po skladbi naprej in nazaj.



Slika 3.8: Približan izsek iz hierarhije. Na sliki je razvidno, da del 30/2 tvorita poddela 7/1 in 8/1, ta dva pa sta sestavljena iz poddelov 1/0, 2/0 in 3/0. Povezave do višjega nivoja nakazujejo, da del 30/2 nastopa kot poddel v več delih na višjem nivoju.



Slika 3.9: Primer vizualizacije aktivacij. Prikazane so tri aktivacije dela s Slike 3.7. Med posameznimi aktivacijami preklapljamo s tipkama + in -, medtem ko z miško stojimo na pripadajočem delu.

3.2.4 Vizualizacija aktivacij

V vozlišču, ki prikazuje posamezen del, podatek v četrti vrstici označuje zaporedno številko trenutno prikazane oziroma nazadnje prikazane aktivacije in število vseh aktivacij (na primer na Sliki 3.7 oznaka 0/3 pomeni, da je trenutno prikazana aktivacija z zaporedno številko 0 od skupno treh aktivacij). Ko se z miško postavimo na enega od delov, se izriše prva aktivacija tega dela: okno s prikazom skladbe se samodejno premakne tako, da prikazuje mesto aktivacije, pravokotniki, ki označujejo tone aktivacije, se obarvajo rdeče, v modrem okvirčku na dnu okna pa se izpišejo višine tonov aktivacije in časi nastopov tonov. Na Sliki 3.9 so prikazane tri aktivacije dela s Slike 3.7.

Ko z miško stojimo na delu, s tipkama + in – preklapljamo med prikazanimi aktivacijami. Pri tem se posodabljajo podatki o lokacijah in časih

Slika 3.10: Izris iskanih vzorcev za lažje testiranje modela. Črni pravokotniki prikazujejo tone iz skladbe, modri okvirčki tone, ki sestavljajo iskani vzorec, rdeči pravokotniki pa tone, ki sestavljajo aktivacijo dela, na katerem trenutno stojimo z miško.

aktivacij ter izris aktivacije na prikazu skladbe.

3.3 Testiranje

Začetno verzijo modela smo testirali s pomočjo vizualizacije na eni skladbi naenkrat in postopno izvajali korake učenja. Opazovali smo naučene dele in aktivacije po posameznih nivojih ter jih primerjali z vzorci, ki smo jih želeli dobiti. V kodi smo iskali vzroke, zakaj se model nekaterih vzorcev ni naučil, in jo sproti dopolnjevali.

Za lažjo primerjavo dobljenih rezultatov z želenimi smo v vizualizacijo na prikaz celotne skladbe dodali izris iskanih vzorcev. Vzorce smo označili z modrimi okvirčki okrog pravokotnikov, ki prikazujejo posamezne tone. Na ta način smo lahko hitro identificirali dele, katerih aktivacije so predstavljale iskani vzorec oziroma del vzorca, pa tudi dele in aktivacije, ki niso pripadale nobenemu želenemu vzorcu. Primer izrisa iskanih vzorcev je prikazan na Sliki 3.10.

V vizualizacijo modela smo dodali tudi funkcionalnost za kreiranje vizualne predstavitve ekspertne anotacije in najdenih vzorcev v programskem orodju Microsoft Excel. Z ročnim pregledovanjem Excelovih datotek smo dobljene rezultate lažje primerjali z želenimi in odkrili problematične dele. Programska oprema Office ali OpenOffice je javno dostopna, zato je format primeren tudi za shranjevanje in izmenjavo datotek z glasbenimi eksperti, ki

nimajo nameščenih namenskih orodij za analizo.

Poglavje 4

Evalvacija rezultatov

Od leta 2013 na dogodku MIREX[1] poteka naloga iskanja ponavljajočih tem in vzorcev (angl. *Discovery of Repeated Themes and Sections*) [10], na kateri smo v letu 2015 sodelovali s kompozicionalnim hierarhičnim modelom za simbolni glasbeni zapis. Za to nalogo sta na voljo dve zbirki podatkov: razvojna, ki je javno objavljena, in testna, na kateri algoritme poganja organizator. V nadaljevanju predstavljamo obe bazi in rezultate kompozicionalnega hierarhičnega modela za simbolni glasbeni zapis na obeh bazah ter jih primerjamo z rezultati ostalih udeležencev.

4.1 Bazi skladb JKUPDD in JKUPTD

Raziskovalci na oddelku za računalniško zaznavanje univerze Johannes Kepler v Linzu pripravljajo bazo klasičnih skladb z anotiranimi ponavljajočimi vzorci. Ekspertno anotacijo so zasnovali na podlagi naslednjih del:

- Barlow in Morgenstern: Slovar glasbenih tem (1953, angl. *Dictionary of Musical Themes*)
- Schoenberg: Osnove glasbene kompozicije (1967, angl. *Fundamentals of Musical Composition*)

[1]Dogodek za skupno evalvacijo rezultatov s področja pridobivanja informacij iz glasbe. Več v poglavju 2.1.2

- Bruhn: Dobro uglašeni klavir J. S. Bacha: Poglobljena analiza in interpretacija (1993, angl: *J.S.Bach's Well-Tempered Clavier: In-depth Analysis and Interpretation*).

Za namene spodbujanja razvoja algoritmov za iskanje ponavljajočih vzorcev so manjši del baze javno objavili pod imenom JKUPDD (razvojna baza vzorcev univerze Johannes Kepler, angl. *JKU Patterns Development Database*) in je dostopna na [10]. Razvojna baza obsega 5 skladb:

- Bach: Dobro uglašeni klavir II (angl. *The Well-Tempered Clavier II*), Preludij in fuga v a-molu, BWV 889
- Beethoven: Menuet iz Sonate št. 1 v Es-duru
- Chopin: Mazurka v b-molu, opus 24, št. 4
- Gibbons: Silver swan
- Mozart: Menuet iz Sonate št. 4 v Es-duru.

Baza vsebuje štiri variante vsake skladbe: monofonično in polifonično verzijo, vsako podano v simbolnem in avdio zapisu. Skladbe v simbolnem zapisu so podane v treh različnih formatih: MIDI, kern[2] in csv s petimi stolpci: čas nastopa tona (angl. *ontime*), merjen v številu dob od začetka skladbe, MIDI številka tona, višina tona, kot jo definira Meredith[3], trajanje tona, merjeno v številu dob in zaporedna številka takta. Polifonične skladbe z označenimi glasovi v večglasju so v monofonične pretvorjene tako, da so glasovi zapisani zaporedno eden za drugim od najvišjega do najnižjega. Polifonične skladbe, ki niso razdeljene na posamezne glasove, pa so v monofonične pretvorjene po principu "clipped skyline approach"– najprej zapišemo melodijo

[2]kern je format za zapis glasbe. Omogoča zapis tonske višine in notnega trajanja, pa tudi drugih informacij, kot so predznaki, taktnice, fraze, smer notnega vratu itd. Format je bolj kot vizualni predstavitvi glasbe namenjen predstavitvi vsebinskih informacij. Zasnovan je bil z namenom olajšanja glasbene analize. [1]

[3]Razložena v poglavju 2.3

iz najvišjih tonov in nato nadaljujemo od zgoraj navzdol do melodije, sestavljene iz najnižjih tonov [10].

Ekspertna anotacija je podana kot dvojica časa nastopa tona in MIDI številke tona. Na začetku je zapisan osnovni vzorec, sledijo pa mu ponovitve vzorca, ki se lahko od osnovnega tudi nekoliko razlikujejo [10].

JKUPTD (testna baza vzorcev univerze Johannes Kepler, angl. *JKU Patterns Test Database*) je baza, ki se uporablja za evalvacijo rezultatov na dogodku MIREX. Vsebuje 5 skladb, ki niso javno objavljene, organizirana pa je enako kot razvojna baza JKUPDD.

4.2 Mere za evalvacijo rezultatov

Povzeto po [10].

Za ocenjevanje uspešnosti algoritma smo uporabili naslednje mere:

- Standardna natančnost, standardni priklic in standardna mera F1
- Vzpostavitvena natančnost, vzpostavitveni priklic in vzpostavitvena mera F1 (angl. *establishment precision, establishment recall, and establishment F1 score*)
- Pojavitvena natančnost, pojavitveni priklic in pojavitvena mera F1 (angl. *occurrence precision, occurrence recall, and occurrence F1 score*)
- Tronivojska natančnost, tronivojski priklic in tronivojska mera F1 (angl. *three-layer precision, three-layer recall, and three-layer F1 score*)

4.2.1 Standardna natančnost, standardni priklic in standardna mera F1

Označimo z n_P število vzorcev v ekspertni anotaciji, z n_Q število vzorcev, ki jih vrne algoritem in s k število vzorcev iz ekspertne anotacije, ki jih je algoritem našel. Standardno natančnost P definiramo kot

$$P = \frac{k}{n_Q}, \tag{4.1}$$

standardni priklic R kot

$$R = \frac{k}{n_P} \tag{4.2}$$

in standardno mero $F1$ kot

$$F1 = \frac{2PR}{(P+R)}. \tag{4.3}$$

Standardne mere v primeru, da se prepoznani vzorec samo za eno noto razlikuje od vzorca iz ekspertne anotacije, le-tega ne štejejo kot uspešno prepoznanega. Zato so bile za dogodek MIREX predlagane nove mere, ki so robustne in dopuščajo manjša odstopanja med rezultatom algoritma in ekspertno anotacijo. Pri evalvaciji rezultatov želimo oceniti, ali je algoritem prepoznal vsaj eno pojavitev posameznega vzorca in kolikšen delež ponovitev posameznega vzorca je našel. Za ta namen se uporabljajo spodaj predstavljene mere, izpeljane iz matrike ocen.

4.2.2 Matrika ocen

Predpostavimo, da imamo v ekspertni anotaciji vzorec P s pojavitvami $P = \{P_1, P_2, \ldots, P_{m_p}\}$ in vzorec Q, ki ga vrne algoritem, s pojavitvami $Q = \{Q_1, Q_2, \ldots, Q_{m_q}\}$. Za vsak P_i in Q_j lahko izmerimo njuno podobnost z uporabo kardinalnosti, ki je za glasbo v simbolnem zapisu podana kot

$$s_c(P_i, Q_j) = \frac{\left|P_i \cap Q_j\right|}{\max\left\{\left|P_i\right|, \left|Q_j\right|\right\}}. \tag{4.4}$$

Matrika ocen vsebuje kardinalnosti za vse pare P_i in Q_j:

$$s(P,Q) = \begin{pmatrix} s(P_1,Q_1) & s(P_1,Q_2) & \cdots & s(P_1,Q_{m_Q}) \\ s(P_2,Q_1) & s(P_2,Q_2) & \cdots & s(P_2,Q_{m_Q}) \\ \vdots & \vdots & \ddots & \vdots \\ s(P_{m_P},Q_1) & s(P_{m_P},Q_2) & \cdots & s(P_{m_P},Q_{m_Q}) \end{pmatrix} \tag{4.5}$$

4.2.3 Vzpostavitvena natančnost, vzpostavitveni priklic in vzpostavitvena mera F1

Če nas za nek algoritem ne zanima, ali je algoritem našel vse pojavitve določenega vzorca, ampak želimo vedeti le, če je našel vsaj eno pojavitev vzorca P v skladbi, uporabimo vzpostavitvene mere. Ustrezen podatek o tem nam da element matrike ocen z najvišjo vrednostjo, ki ga označimo s $S(P, Q)$.

Za skladbo z anotiranimi vzorci $\Pi = \{P_1, P_2, \ldots, P_{n_P}\}$ in vzorci, ki jih vrne algoritem $\Xi = \{Q_1, Q_2, \ldots, Q_{n_Q}\}$, lahko merimo zmožnost algoritma za prepoznavo vsaj ene pojavitve posameznega vzorca s pomočjo vzpostavitvene matrike:

$$S(\Pi, \Xi) = \begin{pmatrix} S(P_1, Q_1) & S(P_1, Q_2) & \cdots & S(P_1, Q_{n_Q}) \\ S(P_2, Q_1) & S(P_2, Q_2) & \cdots & S(P_2, Q_{n_Q}) \\ \vdots & \vdots & \ddots & \vdots \\ S(P_{n_P}, Q_1) & S(P_{n_P}, Q_2) & \cdots & S(P_{n_P}, Q_{n_Q}) \end{pmatrix} \tag{4.6}$$

Vzpostavitveno natančnost P_{est} (angl. *Establishment Precision*) izračunamo iz vzpostavitvene matrike po naslednji formuli:

$$P_{est} = \frac{1}{n_Q} \sum_{j=1}^{n_Q} \max\{S(P_i, Q_j) | i = 1, \ldots, n_P\}. \tag{4.7}$$

Če algoritem v celoti najde k vzorcev in preostalih $n_Q - k$ popolnoma zgreši, je natančnost vzpostavitve enaka standardni natančnosti (k/n_Q).

Vzpostavitveni priklic R_{est} (angl. *Establishment Recall*) izračunamo iz vzpostavitvene matrike po formuli

$$R_{est} = \frac{1}{n_P} \sum_{i=1}^{n_P} \max\{S(P_i, Q_j) | j = 1, \ldots, n_Q\}. \tag{4.8}$$

Vzpostavitvena mera F1 – $F1_{est}$ (angl. *Establishment F1 Score*) se izračuna enako kot standardna mera F1, le da standardno natančnost zamenjamo z

vzpostavitveno natančnostjo in standardni priklic z vzpostavitvenim priklicem:

$$F1_{est} = \frac{2P_{est}R_{est}}{P_{est} + R_{est}}. \tag{4.9}$$

4.2.4 Pojavitvena natančnost, pojavitveni priklic in pojavitvena mera F1

Pojavitvene mere se osredotočajo na sposobnost algoritma najti vse pojavitve nekega vzorca. Te mere visoko ocenijo algoritem, ki odkrije vse pojavitve najdenega vzorca, čeprav popolnoma zgreši veliko drugih pomembnih vzorcev v skladbi.

Indeksi I elementov vzpostavitvene matrike z vrednostmi, večjimi ali enakimi od mejne vrednosti c (privzeta vrednost je 0,75), označujejo, kateri anotirani vzorci se štejejo kot uspešno prepoznani. S pomočjo teh indeksov definiramo pojavitveno matriko. Začnemo z ničelno matriko velikosti $n_P \times n_Q$. Nato za vsak par $(i, j) \in I$ izračunamo natančnost elementa $s(P_i, Q_j)$ iz matrike ocen in izračunano vrednost zapišemo v element (i, j) pojavitvene matrike.

Pojavitvena natančnost *Pocc* (angl. *Occurrence Precision*) je definirana kot natančnost pojavitvene matrike s seštevanjem po neničelnih stolpcih. Pojavitveni priklic *Rocc* (angl. *Occurrence Recall*) je definiran kot priklic pojavitvene matrike s seštevanjem po neničelnih vrsticah.

4.2.5 Tronivojska natančnost, tronivojski priklic in tronivojska mera F1

Tronivojska natančnost P_3 (angl. *Three-Layer Precision*), tronivojski priklic R_3 (angl. *Three-Layer Recall*) in tronivojska mera F1 - F_3 (angl. *Three-Layer F1 Score*) predstavljajo kompromis med vzpostavitvenimi in pojavitvenimi

merami. Hkrati ocenjujejo zmožnost algoritma za prepoznavo čim več pojavitev čim več različnih vzorcev. Definirane so s spodnjimi enačbami:

$$F_3(\Pi, \Xi) = \frac{2P_3(\Pi, \Xi)R_3(\Pi, \Xi)}{P_3(\Pi, \Xi) + R_3(\Pi, \Xi)}, \tag{4.10}$$

kjer je

$$P_3(\Pi, \Xi) = \frac{1}{n_q} \sum_{j=1}^{n_Q} \max\{F_2(P_i, Q_j)|i = 1, \ldots, n_P\}, \tag{4.11}$$

$$R_3(\Pi, \Xi) = \frac{1}{n_p} \sum_{i=1}^{n_P} \max\{F_2(P_i, Q_j)|j = 1, \ldots, n_Q\}, \tag{4.12}$$

$$F_2(P, Q) = \frac{2P_2(P, Q)R_2(P, Q)}{P_2(P, Q) + R_2(P, Q)} \tag{4.13}$$

$$P_2(P, Q) = \frac{1}{m_q} \sum_{l=1}^{m_Q} \max\{F_1(P_k, Q_l)|k = 1, \ldots, m_P\}, \tag{4.14}$$

$$R_2(P, Q) = \frac{1}{m_p} \sum_{k=1}^{m_P} \max\{F_1(P_k, Q_l)|l = 1, \ldots, m_Q\}, \tag{4.15}$$

$$F_1(P, Q) = \frac{2P_1(P, Q)R_1(P, Q)}{P_1(P, Q) + R_1(P, Q)}, \tag{4.16}$$

$$P_1(P, Q) = \frac{|P \cap Q|}{|Q|}, \tag{4.17}$$

$$R_1(P, Q) = \frac{|P \cap Q|}{|P|}. \tag{4.18}$$

4.3 Določitev parametrov

Iskanje optimalnih parametrov je potekalo na razvojni bazi JKUPDD. V modelu lahko spreminjamo veliko število parametrov. Nekatere smo pustili na privzetih vrednostih za avdio CHM, za druge smo ročno izbrali nekaj vrednosti in primerjali rezultate, bolj pa smo se osredotočili na parametra za faktor halucinacije τ_H in faktor inhibicije τ_I, razložena v podpoglavju 3.1, ki sta najbolj vplivala na rezultat. Oba parametra zavzemata vrednosti na intervalu $[0, 1]$ in sta določena za vsak nivo posebej. Model smo prilagodili

tako, da smo za vsakega od parametrov podali vektor vrednosti, ki smo jih želeli preizkusiti, model pa je za vsako kombinacijo parametrov poiskal vzorce in vrnil rezultat.

Ker je generiranje hierarhije in inferenca časovno zelo zahteven proces, smo za računanje vsakokrat izbrali 5 vrednosti za vsakega od parametrov, ki smo jih želeli preizkusiti. Glede na rezultat smo nato naslednjih 5 vrednosti izbrali na manjšem intervalu bližje vrednostim, pri katerih smo dobili boljše rezultate. Začeli smo z enakimi parametri za vse nivoje, nato pa smo jih prilagajali za vsak nivo posebej.

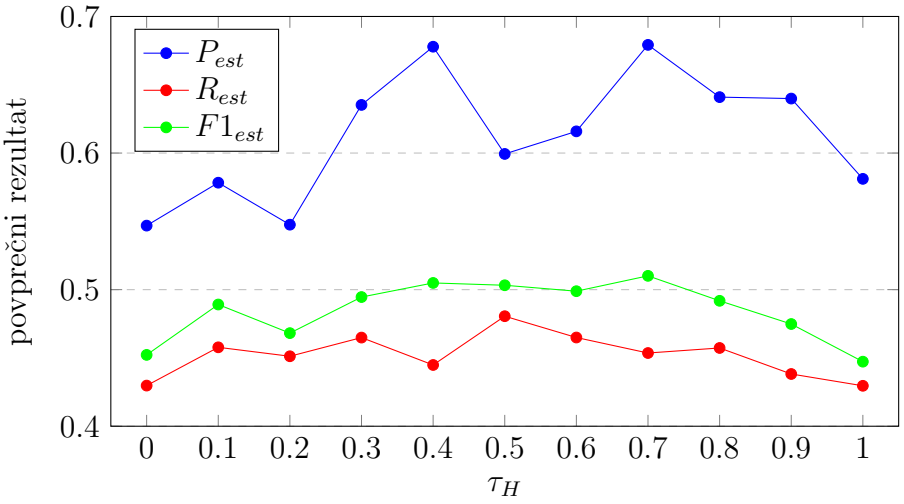
Za ocenjevanje uspešnosti rezultatov smo uporabili mere, opisane v poglavju 4.2. Za izračun mer smo uporabili skripto, dostopno na [10], ter jo prilagodili tako, da je prebrala vse datoteke z vzorci znotraj izbranega direktorija in zapisala vrednosti mer po skladbah v en dokument, grupirano po parametrih. Za izbor najboljših parametrov izmed vseh upoštevanih smo napisali novo skripto, v kateri smo si za vsako od mer $F1_{est}$, $F1_{occ}(c = 0,75)$, F_3 in $F1_{occ}(c = 0,5)$ shranili najboljših 5 parametrov in nato v grafu izrisali unijo vseh štirih množic parametrov. S pregledom grafa smo ročno izbrali parametre za naslednjo iteracijo.

Najboljše rezultate modela smo dobili pri enakih vrednostih posameznega parametra za vse nivoje. V rezultatih, predstavljenih v poglavju 4.4, smo za vse nivoje uporabili vrednosti parametra $\tau_H = 0,7$ in $\tau_I = 0,2$. Obnašanje modela pri različnih parametrih je prikazano v Slikah 4.1- 4.5.

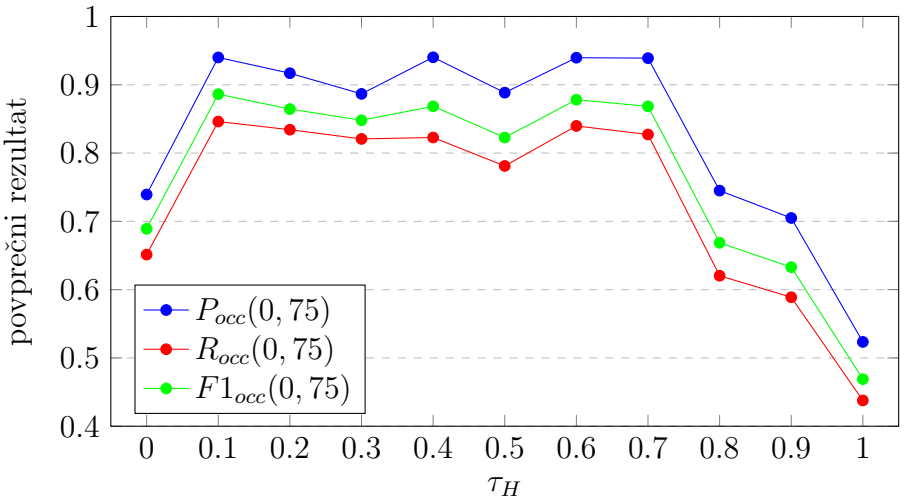
Primerjava pojavitvenih mer pri fiksni vrednosti $\tau_I = 0,2$ in različnih vrednostih τ_H na Sliki 4.1 pokaže, da je pojavitvena natančnost najboljša pri vrednostih $\tau_H = 0,4$ in $\tau_H = 0,7$, vendar je pojavitveni priklic pri $\tau_H = 0,4$ nekoliko slabši, zato je pojavitvena mera F1 najboljša pri vrednosti $\tau_H = 0,7$.

Vzpostavitvene mere pri parametru $c = 0,75$ in fiksni vrednosti $\tau_I = 0,2$ in različnih vrednostih τ_H, prikazane na Sliki 4.2, imajo zelo podobne rezultate pri $\tau_H = 0,1$, $\tau_H = 0,4$, $\tau_H = 0,6$ in $\tau_H = 0,7$.

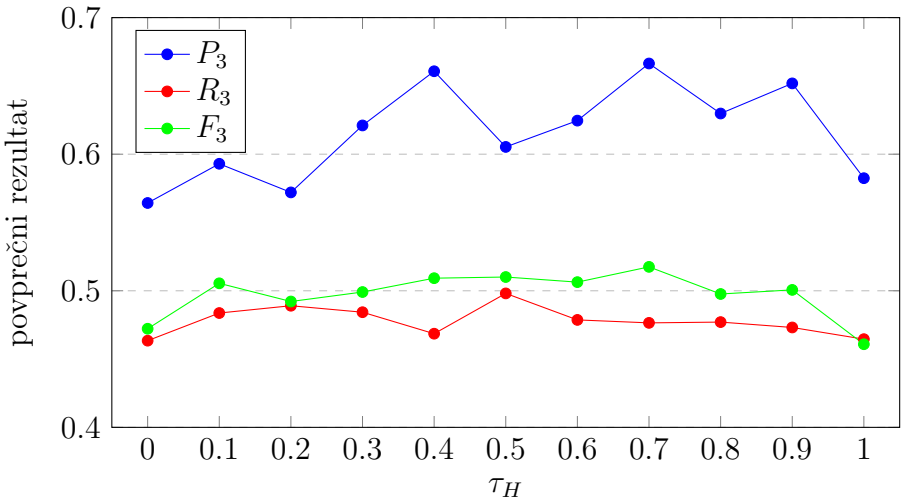
Primerjava tronivojskih mer pri fiksni vrednosti $\tau_I = 0,2$ in različnih vrednostih τ_H na Sliki 4.3 kaže podoben trend kot primerjava vzpostavitve-



Slika 4.1: Primerjava vzpostavitvenih mer na bazi JKUPDD pri različnih vrednostih parametra τ_H in fiksni vrednosti $\tau_I = 0,2$.



Slika 4.2: Primerjava pojavitvenih mer s parametrom $c = 0,75$ na bazi JKUPDD pri različnih vrednostih parametra τ_H in fiksni vrednosti $\tau_I = 0,2$.



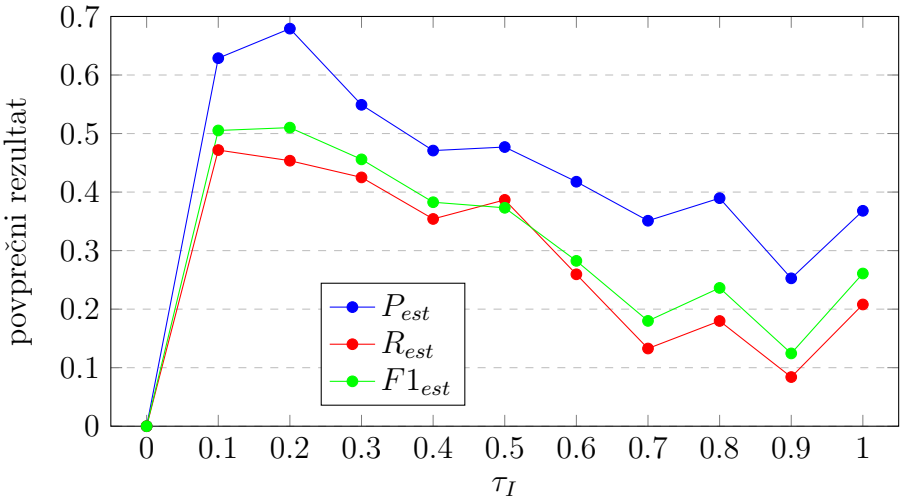
Slika 4.3: Primerjava tronivojskih mer na bazi JKUPDD pri različnih vrednostih parametra τ_H in fiksni vrednosti $\tau_I = 0,2$.

nih mer. Tronivojska natančnost je najboljša pri vrednostih $\tau_H = 0,4$ in $\tau_H = 0,7$, vendar je tronivojski priklic pri $\tau_H = 0,4$ nekoliko slabši, zato je tronivojska mera F1 najboljša pri vrednosti $\tau_H = 0,7$.

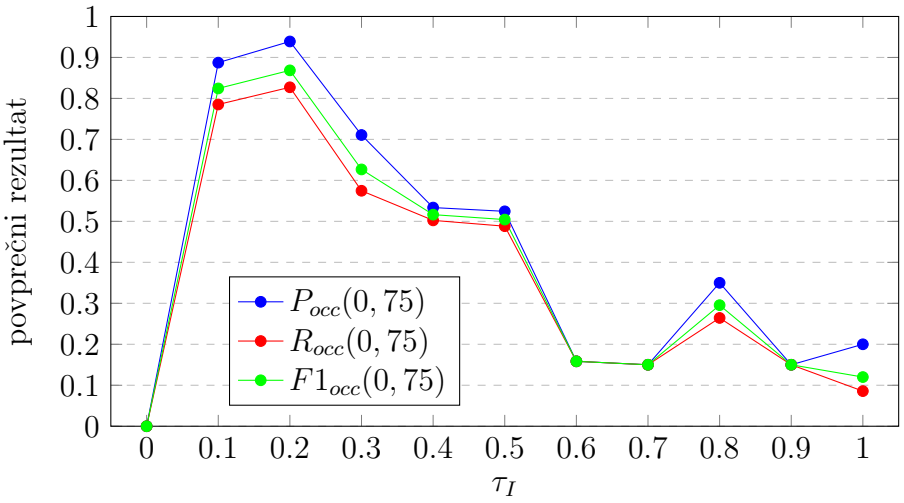
Na Sliki 4.4 vidimo, da je pri fiksni vrednosti $\tau_H = 0,7$ in različnih vrednostih τ_I pojavitvena natančnost najboljša pri $\tau_I = 0,2$, pojavitveni priklic je sicer malo boljši pri $\tau_I = 0,1$, zato je pojavitvena mera F1 približno enaka pri parametrih $\tau_I = 0,1$ in $\tau_I = 0,2$.

Vse tri pojavitvene mere pri fiksni vrednosti $\tau_H = 0,7$ in različnih vrednostih τ_I, prikazane na Sliki 4.5, so najboljše pri $\tau_I = 0,2$.

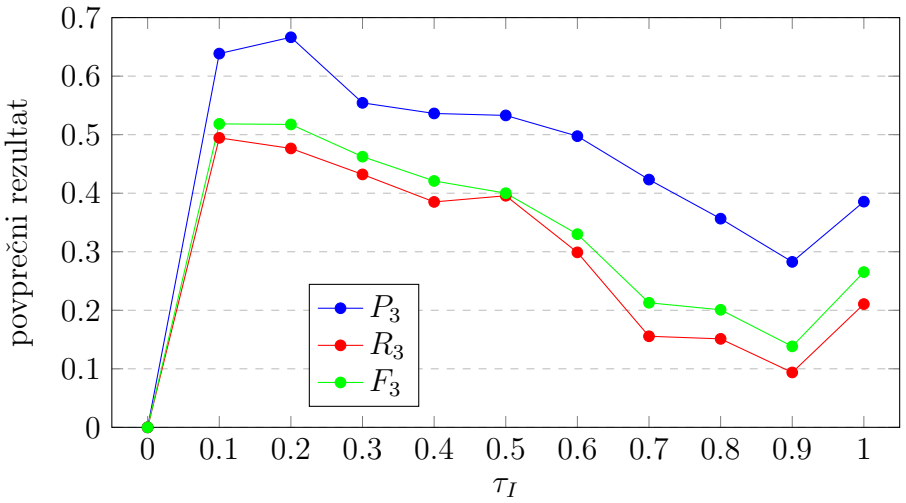
Primerjava tronivojskih mer na Sliki 4.6 zopet kaže podoben trend kot primerjava vzpostavitvenih mer – pojavitvena natančnost je najboljša pri $\tau_I = 0,2$, pojavitveni priklic boljši pri $\tau_I = 0,1$, pojavitvena mera F1 pa približno enaka pri parametrih $\tau_I = 0,1$ in $\tau_I = 0,2$.



Slika 4.4: Primerjava vzpostavitvenih mer na bazi JKUPDD pri različnih vrednostih parametra τ_I in fiksni vrednosti $\tau_H = 0,7$.



Slika 4.5: Primerjava pojavitvenih mer s parametrom $c = 0,75$ na bazi JKUPDD pri različnih vrednostih parametra τ_I in fiksni vrednosti $\tau_H = 0,7$.



Slika 4.6: Primerjava tronivojskih mer na bazi JKUPDD pri različnih vrednostih parametra τ_I in fiksni vrednosti $\tau_H = 0,7$.

4.4 Rezultati

V Tabeli 4.1 so predstavljeni rezultati algoritma SymCHM na bazi JKUPDD, v Tabeli 4.2 pa rezultati na bazi JKUPTD. Oznaka n_P predstavlja število vzorcev v ekspertni anotaciji, n_Q pa število vzorcev, ki jih je našel algoritem. Ostale oznake predstavljajo mere, opisane v poglavju Mere za evalvacijo rezultatov.

mera/skladba	Bach	Beethoven	Chopin	Gibbons	Mozart	povprečje
n_P	3	7	4	8	9	/
n_Q	2	5	14	3	12	/
P_{est}	0,7857	0,7860	0,4759	0,9425	0,4058	0,6792
R_{est}	0,3333	0,6086	0,6200	0,2950	0,4112	0,4536
$F1_{est}$	0,4681	0,6860	0,5385	0,4494	0,4085	0,5101
$P_{occ}(c = 0,75)$	1,0000	0,8429	0,9095	0,9425	1,0000	0,9390
$R_{occ}(c = 0,75)$	0,5714	0,8633	0,7015	1,0000	1,0000	0,8272
$F1_{occ}(c = 0,75)$	0,7273	0,8530	0,7921	0,9704	1,0000	0,8685
P_3	0,5522	0,8463	0,4913	0,9425	0,5000	0,6664
R_3	0,2424	0,6631	0,6922	0,2777	0,5070	0,4765
F_3	0,3369	0,7436	0,5747	0,4290	0,5035	0,5175
$P_{occ}(c = 0,5)$	0,7381	0,8127	0,7184	0,9425	0,7146	0,7853
$R_{occ}(c = 0,5)$	0,5714	0,7365	0,6132	1,0000	0,7286	0,7299
$F1_{occ}(c = 0,5)$	0,6442	0,7727	0,6616	0,9704	0,7215	0,7541
P	0,5000	0,0000	0,0000	0,6667	0,0833	0,2500
R	0,3333	0,0000	0,0000	0,2500	0,1111	0,1389
$F1$	0,4000	0,0000	0,0000	0,3636	0,0952	0,1718

Tabela 4.1: Rezultati algoritma SymCHM na bazi JKUPDD

mera/skladba	1	2	3	4	5	povprečje
n_P	5	5	10	8	13	/
n_Q	18	31	8	2	35	/
P_{est}	0,6998	0,2086	0,4255	0,9667	0,3664	0,5334
R_{est}	0,7307	0,4540	0,2770	0,2774	0,3310	0,4140
$F1_{est}$	0,7149	0,2859	0,3356	0,4311	0,3478	0,4231
$P_{occ}(c = 0,75)$	0,7265	0,7545	0,8336	0,9667	0,7852	0,8133
$R_{occ}(c = 0,75)$	0,4187	0,5045	0,6947	0,9667	0,4059	0,5981
$F1_{occ}(c = 0,75)$	0,5312	0,6047	0,7578	0,9667	0,5351	0,6791
P_3	0,3783	0,2220	0,5006	0,9828	0,3540	0,4875
R_3	0,4986	0,4133	0,3055	0,2735	0,3721	0,3726
F_3	0,4302	0,2888	0,3794	0,4279	0,3628	0,3778
$P_{occ}(c = 0,5)$	0,5792	0,6697	0,8336	0,9667	0,6171	0,7333
$R_{occ}(c = 0,5)$	0,4326	0,5727	0,6947	0,9667	0,4563	0,6246
$F1_{occ}(c = 0,5)$	0,4952	0,6174	0,7578	0,9667	0,5247	0,6724
P	0,0000	0,0323	0,0000	0,5000	0,0000	0,1065
R	0,0000	0,2000	0,0000	0,1250	0,0000	0,0650
$F1$	0,0000	0,0556	0,0000	0,2000	0,0000	0,0511

Tabela 4.2: Rezultati algoritma SymCHM na bazi JKUPTD [11]

4.5 Diskusija

Rezultati na razvojni bazi so pričakovano boljši od rezultatov na testni bazi, saj smo na razvojni bazi algoritem večkrat poganjali z različnimi vrednostmi parametrov.

Pri pregledu števila najdenih vzorcev n_Q in števila vzorcev iz ekspertne anotacije n_P opazimo, da pri nekaterih skladbah najdemo premalo vzorcev, pri drugih pa občutno preveč. Skladbe, pri katerih najdemo premalo vzorcev, v ekspertni anotaciji vsebujejo zelo kratke vzorce, ki jih zaradi omejitve rezultata na vzorce od četrtega nivoja navzgor izpustimo. Če smo v rezultat vključili tudi vzorce s tretjega nivoja, smo dobili veliko kratkih vzorcev, ki se ne smatrajo za glasbeno pomembne in s tem poslabšali rezultat. Za iskanje kratkih relevantnih vzorcev bi morali v model vključiti logiko, ki bi znala ločiti med pomembnimi in nepomembnimi vzorci, kot jih dojema poslušalec oziroma glasbeni analitik. Trenutna verzija modela temelji samo na statističnih pojavitvah vzorcev in ne vključuje glasbenega predznanja, zato glasbeno signifikantnih vzorcev ne zna izločiti. Kot dolžino vzorca v modelu upoštevamo število tonov, ki jih vzorec zajema. Morda bi kratke vzorce zajeli z uteževanjem vzorcev glede na njihovo absolutno dolžino (glede na čas trajanja tonov vzorca) in pripisovanjem večje uteži osamljenim vzorcem (vzorcem, ki so obdani s pavzami in ne z drugimi toni).

Skladbe, pri katerih najdemo preveč vzorcev, vsebujejo zelo dolge vzorce ali ponovljene celotne odseke skladbe (odseke, ki so v notnem črtovju označeni s ponavljajem in se jih zaigra dvakrat). Pri zelo dolgih vzorcih (na primer 150 tonov pri Chopinu) ne zajamemo celotnega vzorca zaradi omejitve rezultata na 6 nivojev, saj na šestem nivoju zajamemo največ $2^6 = 64$ tonov. Pri vzorcih, ki niso daljši od 64 tonov, pa namesto celotnega dolgega vzorca pogosto najdemo več krajših vzorcev, ki se nahajajo znotraj daljšega. Pri učenju namreč preferiramo tiste dele, ki pokrijejo večji del učne množice. Če dolg vzorec razbijemo na več manjših in se eden od manjših vzorcev v daljšem ponovi večkrat, drugi pa samo enkrat, lahko slednjega v postopku učenja izpustimo in se posledično nikoli ne naučimo celotnega dolgega vzorca.

Še en vzrok za veliko število vzorcev je podvajanje nekaterih vzorcev, ki so skoraj enaki, ampak ima eden od njih na primer en ton več kot drugi. Večkratno pokritje istih tonov iz vhodnih podatkov je posledica nizke izbrane vrednosti parametra inhibicije ($\tau_I = 0,2$), ki nam dovoljuje, da več delov modela pokrije isti odsek skladbe. Takšne redundantne vzorce bi lahko odstranili z izboljšanim postopkom izbora vzorcev.

Ekspertna anotacija v zbirkah JKUPDD in JKUPTD poleg anotiranih vzorcev vsebuje tudi anotirane ponovljene odseke skladbe (odseke, ki so v notnem črtovju označeni s ponavljajem in se jih zaigra dvakrat zapored). V našem modelu vzorec vključimo v izhodno datoteko samo, če se ponovi vsaj trikrat. Ponovljeni odseki imajo pogosto samo eno ponovitev, zato jih pri izpisu rezultata izpustimo. Ker so ponovljeni odseki običajno daljši od kratkih motivov, ki se ponovijo večkrat v skladbi, bi jih lahko poskusili zajeti tako, da bi povečali število nivojev, ki jih generira model, in mejo minimalnega števila ponovitev za upoštevanje vzorca določili po nivojih – na nižjih nivojih bi bila ta meja višja, po nivojih navzgor pa bi se postopoma zmanjševala. Na ta način bi lahko poskusili vključiti tudi tretji nivo in mejo minimalnega števila ponovitev za upoštevanje vzorca za ta nivo.

Primerjava vrednosti posameznih mer iz Tabel 4.1 in 4.2 nakazuje, da algoritem bolje rešuje problem iskanja vseh ponovitev enega vzorca v skladbi kot iskanja vsaj ene ponovitve vsakega od vzorcev (vrednosti vzpostavitvenih mer so nižje od vrednosti pojavitvenih mer). Nekaj možnih rešitev za izboljšanje deleža odkritih vzorcev smo predhodno že izpostavili v diskusiji. Pri pojavitvenih merah s parametrom $c = 0,75$ dosegamo visoke vrednosti, pri pojavitvenih merah s parametrom $c = 0,5$ pa nekoliko nižje. Parameter c označuje delež ponovitev vzorca, ki jih moramo najti, da se vzorec šteje kot prepoznan. Znižanje pojavitvenih mer ob znižanju parametra c nam pove, da ne najdemo vedno vseh ponovitev vzorca, ampak obstajajo tudi taki, za katere najdemo več kot 50 % in manj kot 75 % ponovitev.

Pri standardnih merah dosegamo nizke vrednosti, kar pomeni, da algoritem ni tako zelo uspešen v iskanju celotnih vzorcev iz ekspertne anotacije

do tona natančno in z vsemi ponovitvami. Najvišji vrednosti dosegamo pri standardni natančnosti na Bachovi in Gibbonsovi skladbi, ki imata v primerjavi z ostalimi tremi skladbami z baze JKUPDD zelo enakomerne vzorce (vzorce pogosto sestavljajo toni enakih notnih trajanj). Nasprotno imamo pri Beethovnovi in Chopinovi skladbi, ki imata zelo razgiban ritem, najslabšo standardno natančnost. To je lahko posledica uteževanja aktivacij v postopku računanja moči aktivacije, v katerem poleg oddaljenosti aktivacij in vmesnih izpuščenih tonov uteži računamo tudi na podlagi enakomernosti notnih trajanj vsebovanih tonov. Vrednosti standardnega priklica za vse skladbe so nizke, torej bolj malo vzorcev iz ekspertne anotacije najdemo v celoti in z vsemi ponovitvami.

Pri večini skladb za vse skupine mer (vzpostavitvene, pojavitvene, tronivojske in standardne) dosegamo višje vrednosti natančnosti kot priklica. Naš model torej v veliki meri vrne relevantne vzorce (večina vzorcev iz rezultata modela se nahaja v ekspertni anotaciji), slabši pa je v odkrivanju čim večjega števila vzorcev iz ekspertne anotacije.

4.5.1 Primerjava rezultatov z ostalimi raziskovalci

V nadaljevanju povprečne rezultate algoritma SymCHM primerjamo s povprečnimi rezultati algoritmov ostalih raziskovalcev. Rezultati na razvojni bazi JKUPDD so na voljo za tiste algoritme, za katere so jih avtorji sami objavili, rezultati na bazi JKUPTD pa so javno objavljeni za vse udeležence dogodka MIREX. Tabeli 4.3 in 4.4 ter Slike 4.7– 4.16 prikazujejo povprečne vrednosti mer vseh petih skladb za vsak algoritem. Rezultati ostalih raziskovalcev prikazujejo podoben trend kot rezultati kompozicionalnega hierarhičnega modela – na razvojni bazi so rezultati boljši kot na testni, torej so algoritmi nekoliko prilagojeni testnim podatkom.

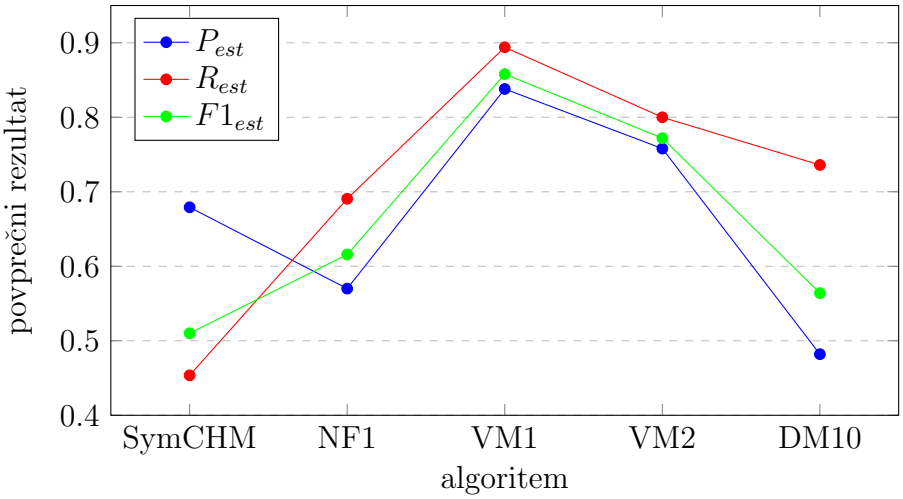
Povprečne vrednosti vzpostavitvenih mer, prikazanih v Slikah 4.7 in 4.8, so za SymCHM nižje od povprečnih vzpostavitvenih mer večine drugih predstavljenih algoritmov. Ostali algoritmi torej najdejo več različnih vzorcev kot SymCHM.

mera/algoritem	SymCHM	NF1	VM1	VM2	DM10
P_{est}	0,6792	0,5700	0,8380	0,7580	0,4820
R_{est}	0,4536	0,6906	0,8940	0,8000	0,7360
$F1_{est}$	0,5101	0,6158	0,8580	0,7720	0,5640
$P_{occ}(c = 0,75)$	0,9390	0,7398	0,7500	0,8180	0,6740
$R_{occ}(c = 0,75)$	0,8272	0,4431	0,8900	0,7840	0,7760
$F1_{occ}(c = 0,75)$	0,8685	0,5398	0,8100	0,7840	0,7120
P_3	0,6664	0,4086	0,6980	0,6640	0,4140
R_3	0,4765	0,5316	0,7480	0,6820	0,6280
F_3	0,5175	0,4544	0,7140	0,6680	0,4720
$P_{occ}(c = 0,5)$	0,7853	0,6205	0,6720	0,7240	0,6880
$R_{occ}(c = 0,5)$	0,7299	0,5006	0,8660	0,7160	0,7480
$F1_{occ}(c = 0,5)$	0,7541	0,5364	0,7520	0,7180	0,7020
P	0,2500	NA	0,3140	0,0280	0,0000
R	0,1389	NA	0,3180	0,0260	0,0000
$F1$	0,1718	NA	0,3080	0,0260	0,0000

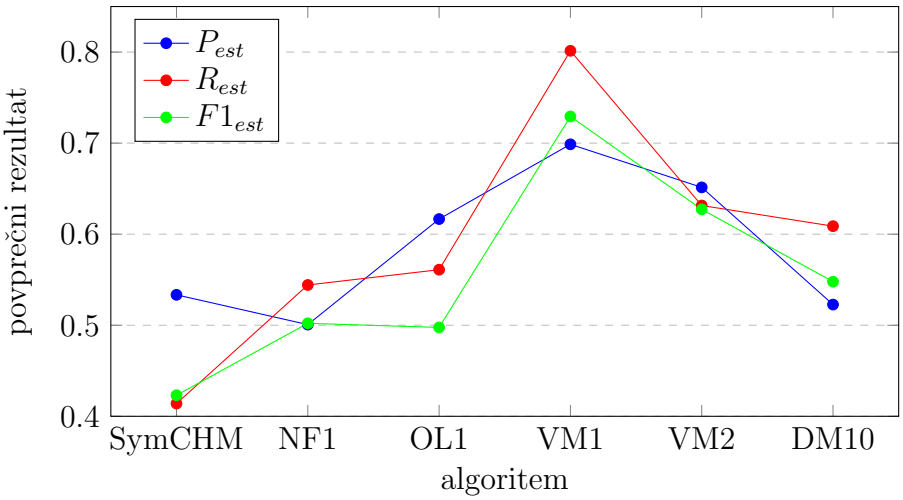
Tabela 4.3: Povprečja rezultatov različnih algoritmov na bazi JKUPDD [42, 51, 37]

mere/algoritem	SymCHM	NF1	OL1	VM1	VM2	DM10
P_{est}	0,5334	0,5007	0,6166	0,6987	0,6514	0,5227
R_{est}	0,4140	0,5442	0,5610	0,8014	0,6315	0,6088
$F1_{est}$	0,4231	0,5021	0,4976	0,7293	0,6273	0,5478
$P_{occ}(c =, 75)$	0,8133	0,5972	0,8790	0,4886	0,6006	0,5667
$R_{occ}(c =, 75)$	0,5981	0,3288	0,7596	0,8057	0,5844	0,7515
$F1_{occ}(c =, 75)$	0,6791	0,4087	0,8066	0,6045	0,5701	0,6242
P_3	0,4875	0,3214	0,5104	0,5431	0,5322	0,4294
R_3	0,3726	0,3891	0,4939	0,4667	0,4208	0,4667
F_3	0,3778	0,3329	0,4275	0,4940	0,4619	0,4326
$P_{occ}(c =, 5)$	0,7333	0,5498	0,7878	0,4487	0,4613	0,4720
$R_{occ}(c =, 5)$	0,6246	0,3342	0,7105	0,7510	0,6097	0,7465
$F1_{occ}(c =, 5)$	0,6724	0,4080	0,7450	0,5585	0,5154	0,5694
P	0,1065	0,0154	0,1601	0,1714	0,0619	0,0267
R	0,0650	0,0500	0,2375	0,1608	0,0650	0,0450
$F1$	0,0511	0,0235	0,1236	0,1600	0,0619	0,0325

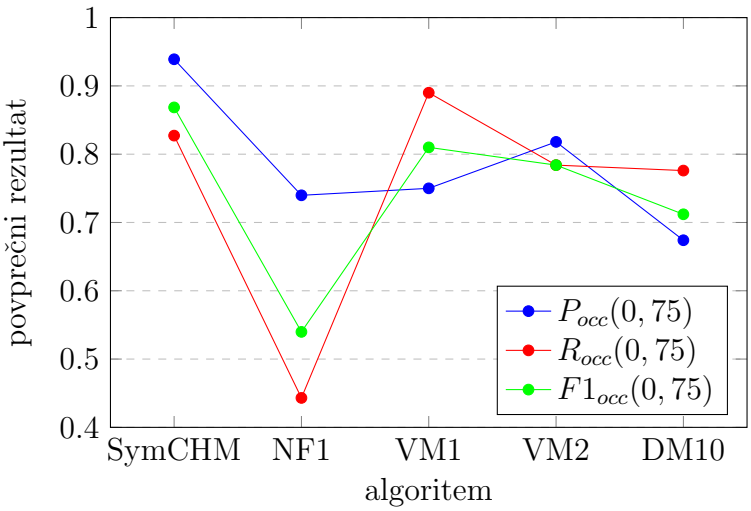
Tabela 4.4: Povprečja rezultatov različnih algoritmov na bazi JKUPTD [8, 11]



Slika 4.7: Primerjava vzpostavitvenih mer na bazi JKUPDD



Slika 4.8: Primerjava vzpostavitvenih mer na bazi JKUPTD



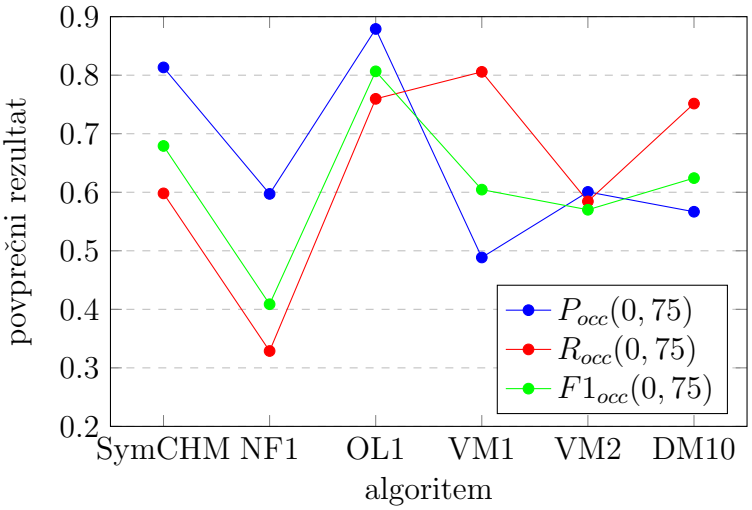
Slika 4.9: Primerjava pojavitvenih mer s parametrom $c = 0,75$ na bazi JKUPDD

V povprečju pojavitvenih mer, prikazanih na Slikah 4.9, 4.10, 4.11 in 4.12, je SymCHM primerljiv ali boljši od ostalih algoritmov. V iskanju čim več ponovitev nekega vzorca smo torej primerljivi ali boljši od ostalih raziskovalcev. Za večino algoritmov so pojavitvene mere pri parametru $c = 0,75$ znatno višje od mer pri parametru $c = 0,5$. Izjema so DM1 na bazi JKUPDD ter NF1 in VM1 na bazi JKUPTD. Iz tega sklepamo, da za večino vzorcev odkrijejo bodisi več kot 75 % ponovitev bodisi manj kot 50 % ponovitev.

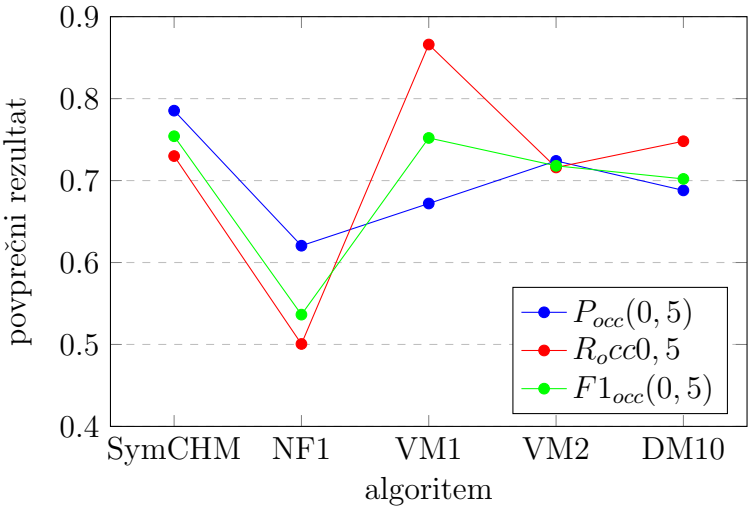
Tronivojske mere predstavljajo kompromis med vzpostavitvenimi in pojavitvenimi merami. Hkrati ocenjujejo zmožnost algoritma za prepoznavo čim več pojavitev čim več različnih vzorcev. Povprečja mer za vse algoritme so prikazana na Slikah 4.13 in 4.14. Po tem kriteriju je SymCHM med slabšimi algoritmi, kar je posledica slabše sposobnosti algoritma za odkrivanje vsaj ene pojavitve vsakega od vzorcev.

Vrednosti standardnih mer so za vse algoritme nizke. Na razvojni bazi SymCHM doseže dobre rezultate pri teh merah, na testni bazi pa je v zlati sredini.

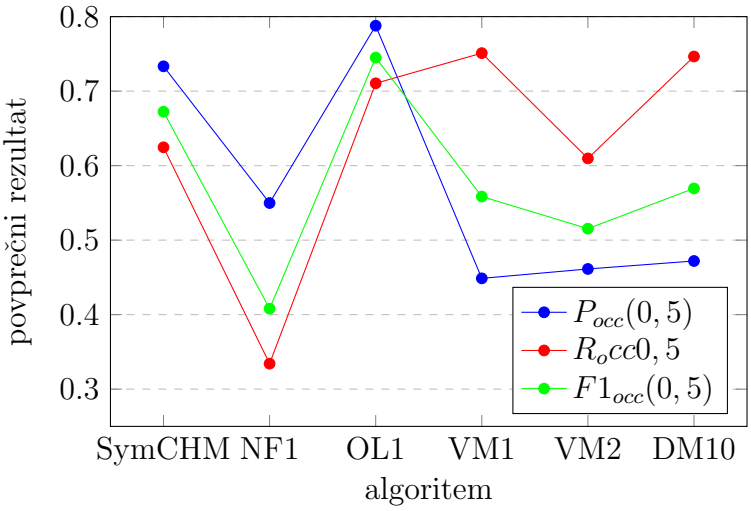
Glede na prikazane rezultate je kompozicionalni hierarhični model za sim-



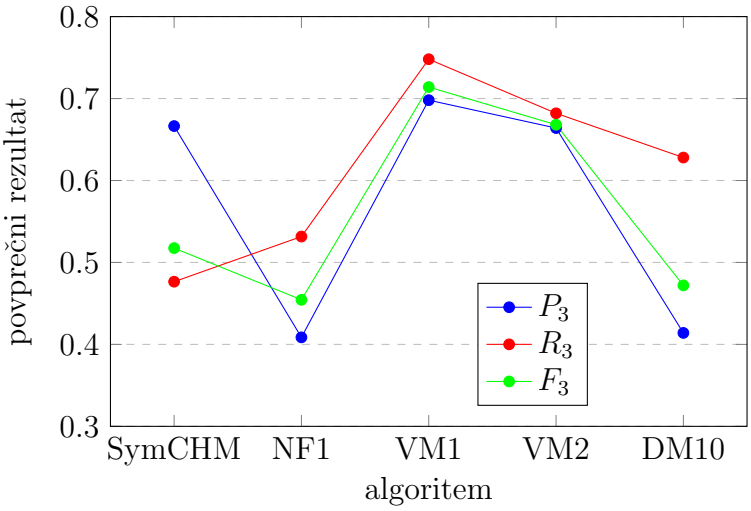
Slika 4.10: Primerjava pojavitvenih mer s parametrom $c = 0,75$ na bazi JKUPTD



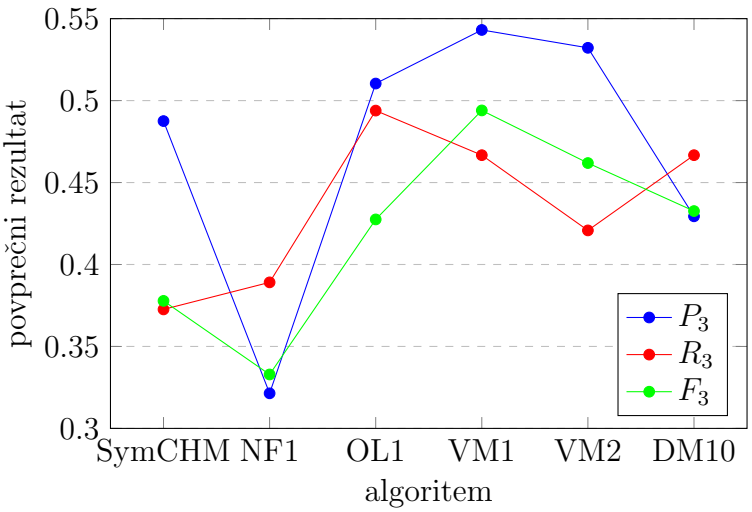
Slika 4.11: Primerjava pojavitvenih mer s parametrom $c = 0,5$ na bazi JKUPDD



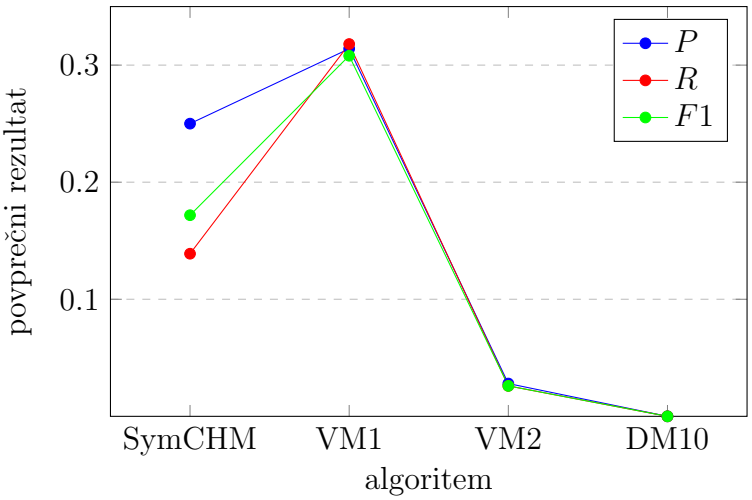
Slika 4.12: Primerjava pojavitvenih mer s parametrom $c = 0,5$ na bazi JKUPTD



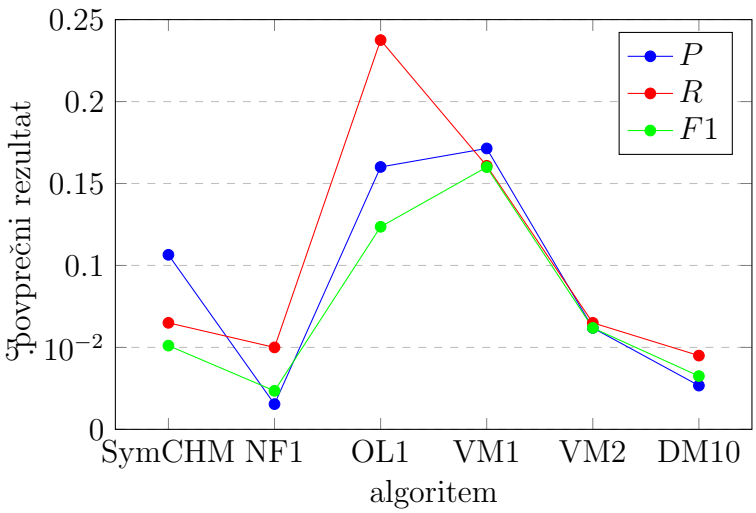
Slika 4.13: Primerjava tronivojskih mer na bazi JKUPDD



Slika 4.14: Primerjava tronivojskih mer na bazi JKUPTD



Slika 4.15: Primerjava standardnih mer na bazi JKUPDD



Slika 4.16: Primerjava standardnih mer na bazi JKUPTD

bolni glasbeni zapis v iskanju vseh ponovitev enega vzorca primerljiv ali boljši od ostalih algoritmov za prepoznavo vzorcev v glasbi, za njimi pa nekoliko zaostaja pri identifikaciji čim več glasbeno relevantnih vzorcev v skladbi. Prostora za izboljšave je še veliko. Nekaj možnih usmeritev za nadaljnje raziskovanje smo podali v prvem delu tega poglavja. Verjamemo, da lahko z nadaljevanjem razvoja izboljšamo svoje rezultate v primerjavi z ostalimi raziskovalci.

Poglavje 5

Sklepne ugotovitve

V nalogi smo predstavili nov pristop k iskanju ponavljajočih vzorcev v glasbi. Uporabili smo obstoječ kompozicionalni hierahični model [43] in ga prilagodili za branje vhodnih podatkov v simbolnem glasbenem zapisu in reševanje naloge iskanja vzorcev. Za lažje testiranje in interpretacijo rezultatov smo implementirali tudi vizualizacijo modela. Rezultate smo evalvirali na dveh zbirkah podatkov in jih primerjali z rezultati ostalih raziskovalcev na istih dveh zbirkah. Primerjava je pokazala, da kompozicionalni hierarhični model za simbolni glasbeni zapis enako dobro ali bolje kot ostali algoritmi najde veliko število ponovitev prepoznanega vzorca, nekoliko pa zaostaja v iskanju vsaj ene pojavitve čim več glasbeno relevantnih vzorcev. Njegove prednosti so:

- *hierarhično strukturiranje pridobljenega znanja*, ki organizira vzorce v nivoje glede na njihovo kompleksnost,
- *transparentnost*, ki omogoča pregled naučenih konceptov in njihovo sestavo iz posameznih manjših gradnikov,
- *robustnost*, ki omogoča ponovno uporabo pridobljenega znanja in apliciranje le-tega na različna mesta v skladbi,
- *možnost apliciranja naučenega modela na poljubne podatke*, ki omogoča iskanje ponavljajočih vzorcev preko več skladb in s tem razširja upo-

rabnost modela tudi na druge sorodne naloge s področja pridobivanja informacij iz glasbe, kot so identifikacija skladatelja, analiza sloga skladatelja, iskanje plagiatov, analiza zbirk narodnih pesmi itd.

5.1 Možnosti za nadaljnji razvoj

Kompozicionalni hierarhični model za simbolni glasbeni zapis odpira številne možnosti za izboljšave in nadaljnji razvoj. Nekaj usmeritev za možne izboljšave smo podali v poglavju 4.5. Večji delež prepoznanih osnovnih vzorcev bi lahko dosegli s povečanjem števila nivojev, ki jih zajamemo pri generiranju rezultata. Trenutno izpišemo nivoje od četrtega do šestega. Z dodajanjem tretjega nivoja se lahko poveča zmožnost prepoznave kratkih vzorcev, vendar je treba formalizirati in implementirati tudi logiko za razločevanje glasbeno pomembnih vzorcev od nepomembnih, da izločimo veliko število nerelevantnih kratkih vzorcev. Za prepoznavo dolgih ponovljenih odsekov bi lahko vključili nivoje od šestega navzgor, vendar bi morali v tem primeru za višje nivoje zmanjšati mejo za število ponovitev, ki jih mora imeti posamezen vzorec, da ga upoštevamo, saj se ponovljeni odseki običajno pojavijo samo dvakrat v skladbi (se samo enkrat ponovijo). Z izboljšanjem postopka filtriranja vzorcev bi lahko zmanjšali delež redundantnih vzorcev v rezultatu. Zaradi velikega števila parametrov, ki določajo obnašanje modela, smo se v okviru naloge osredotočili samo na nekatere. Z optimizacijo vseh parametrov bi se lahko izboljšali tudi rezultati.

Z obstoječim modelom ali njegovo nadgradnjo bi funkcionalnost lahko razširili na reševanje drugih nalog s področja pridobivanja informacij iz glasbe. Model je implementiran tako, da omogoča učenje na eni ali več skladbah naenkrat. V magistrski nalogi smo se omejili na iskanje ponavljajočih vzorcev znotraj ene skladbe, zato smo za vsako skladbo zgradili ločen model. S poganjanjem algoritma na več skladbah naenkrat je mogoče iskati ponavljajoče vzorce, ki se pojavijo v različnih skladbah. To funkcionalnost se lahko izkoristi za študije glasbenega sloga skladatelja ali iskanje plagiatov.

Trenutna implementacija omogoča samo obdelavo monofoničnih skladb, saj predpostavlja, da se ob vsakem času nastopa tona pojavi samo ena tonska višina. S prilagoditvami posameznih procedur bi to omejitev lahko odpravili in funkcionalnost razširili na polifonične skladbe v simbolnem glasbenem zapisu.

Funkcionalnost za prepoznavo akordov in osnovnih frekvenc, ki prebere podatke v avdio zapisu, je implementirana ločeno od funkcionalnosti za prepoznavo vzorcev. Trenutno torej ne omogoča kombinacije branja iz avdio zapisa in posredovanja podatkov algoritmu za prepoznavo vzorcev. Logika za iskanje vzorcev temelji na višinah tonov, ki jih iz simbolnega glasbenega zapisa enostavno preberemo, iz avdio zapisa pa jih je potrebno pridobiti s transkripcijo[1]. Z združevanjem funkcionalnosti spektralnega in simbolnega dela bi lahko dobili model, ki prebere podatke v avdio zapisu, izvede transkripcijo in nato nadaljuje z algoritmom za prepoznavo vzorcev.

[1]Transkripcija je proces transformacije avdio signala v simbolni glasbeni zapis. [44]

Literatura

[1] "Everything You Need to Know About the Humdrum **kern Representation." Dosegljivo: http://www.music-cog.ohio-state.edu/Humdrum/representations/kern.html. [Dostopano: 01.06.2016].

[2] A. Arzt, S. Böck in G. Widmer, "Fast Identification of Piece and Score Position via Symbolic Fingerprinting," v *Proc. of the 13th International Society for Music Information Retrieval Conference*, Porto, Portugal, 2012, str. 433–438.

[3] E. Balaguer-Ballester, N. R. Clark, M. Coath, K. Krumbholz in S. L. Denham, "Understanding pitch perception as a hierarchical process with top-down modulation," *PLoS Computational Biology*, zv. 5, št. 3, 2009.

[4] J. A. Burgoyne, I. Fujinaga in J. S. Downie, "Music Information Retrieval," v *A New Companion to Digital Humanities* (ur. S. Schreibman, R. Siemens, in J. Unsworth), Chichester, UK: John Wiley & Sons, Ltd, 2015, pogl. 15, str. 213–226.

[5] E. Cambouropoulos, "Voice and Stream: Perceptual and Computational Modeling of Voice Separation," *Music Perception*, zv. 26, št. 1, str. 75–94, 2008.

[6] C. Chafe, D. Jaffe, K. Kashima, B. Mont-Reynaud in J. Smith, "Techniques for note identification in polyphonic music," v *Proceedings of the International Computer Music Conference*. San Francisco CA: Computer Music Association, 1985.

[7] E. Chew in X. Wu, "Separating voices in polyphonic music: A contig mapping approach," *Computer Music Modeling and Retrieval*, zv. 3310, str. 1–20, 2005.

[8] T. Collins, "2014: Discovery of Repeated Themes and Sections Results," 2014. Dosegljivo: http://www.music-ir.org/mirex/wiki/2014:Discovery_of_Repeated_Themes_&_Sections. [Dostopano: 01.06.2016].

[9] ——, "2015: Discovery of Repeated Themes and Sections," 2015. Dosegljivo: http://www.tomcollinsresearch.net/mirex-pattern-discovery-task.html. [Dostopano: 01.06.2016].

[10] ——, "2015: Discovery of Repeated Themes and Sections," 2015. Dosegljivo: http://www.music-ir.org/mirex/wiki/2015:Discovery_of_Repeated_Themes_&_Sections. [Dostopano: 01.06.2016].

[11] ——, "2015: Discovery of Repeated Themes and Sections Results," 2015. Dosegljivo: http://www.music-ir.org/mirex/wiki/2015:Discovery_of_Repeated_Themes_&_Sections. [Dostopano: 01.06.2016].

[12] T. Collins, A. Arzt, S. Flossmann in G. Widmer, "SIARCT-CFP: Improving precision and the discovery of inexact musical patterns in point-set representations," v *Proc. of the 14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, 2013, str. 549–554.

[13] T. Collins, S. Böck, F. Krebs in W. G., "Bridging the audio-symbolic gap: the discovery of repeated note content directly from polyphonic music audio," v *Proceedings of the Audio Engineering Society's 53rd Conference on Semantic Audio*, London, UK, 2014.

[14] D. Conklin, "Antipattern discovery in folk tunes," *Journal of New Music Research*, zv. 42, št. 2, str. 161–169, 2013.

[15] J. S. Downie, A. F. Ehmann, M. Bay in M. C. Jones, "The music information retrieval evaluation eXchange: Some observations and insights," *Studies in Computational Intelligence*, zv. 274, str. 93–115, 2010.

[16] M. Farbood, "Working Memory and the Perception of Hierarchical Tonal Structures," v *International Conference on Music Perception and Cognition*, 2010, str. 219–222.

[17] S. Fidler in A. Leonardis, "Towards scalable representations of visual categories: Learning a hierarchy of parts," v *IEEE Computer Vision and Pattern Recognition*, 2007.

[18] J. Forth, "Cognitively-Motivated Geometric Methods of Pattern Discovery and Models of Similarity in Music," Doktorska disertacija, University of London, 2012.

[19] M. Hamanaka, K. Hirata in S. Tojo, "Implementing a Generative Theory of Tonal Music," *Journal of New Music Research*, zv. 35, št. 4, str. 249–277, 2006.

[20] D. Huron, *Sweet Anticipation: Music and the Psychology of Expectation.* MIT Press, 2006.

[21] B. Janssen, W. B. De Haas, A. Volk in P. Van Kranenburg, "Discovering Repeated Patterns in Music: State of Knowledge, Challenges, Perspectives," v *Proc. of the 10th International Symposium on Computer Music Multidisciplinary Research*, Marseille, France, 2013, str. 225–240.

[22] B. Janssen, P. van Kranenburg in A. Volk, "A comparison of symbolic similarity measures for finding occurrences of melodic segments," v *Proc. of the 16th International Society for Music Information Retrieval Conference*, 2015, str. 659–665.

[23] T. Kageyama, K. Mochizuki in T. Yosuke, "Melody Retrieval with Humming," v *Proceedings of the International Computer Music Conference*, Waseda University, Japan, 1993, str. 349–351.

[24] M. Kassler, "Toward Musical Information Retrieval," *Perspectives of New Music*, zv. 4, št. 2, str. 59–67, 1966.

[25] P. B. Kirlin in P. E. Utgoff, "A Framework for Automated Schenkerian Analysis," v *9th International Conference on Music Information Retrieval*, 2008, str. 363–368.

[26] S. Koelsch, M. Rohrmeier, R. Torrecuso in S. Jentschke, "Processing of hierarchical syntactic structure in music," v *Proceedings of the National Academy of Sciences of the United States of America*, 2013.

[27] C. L. Krumhansl in L. L. Cuddy, "A Theory of Tonal Hierarchies in Music," v *Music Perception* (ur. M. R. Jones, R. R. Fay in A. N. Popper), New York, 2010, pogl. 3.

[28] O. Lartillot, "Multi-dimensional motivic pattern extraction founded on adaptive redundancy filtering," *Journal of New Music Research*, zv. 34, št. 4, str. 375–393, 2005.

[29] ——, "In-depth Motivic Analysis Based on Multiparametric Closed Pattern and Cyclic Sequence Mining," v *Proc. of the 15th International Society for Music Information Retrieval Conference*, Taipei, Taiwan, 2014, str. 361–366.

[30] F. Lerdahl, "Genesis and Architecture of GTTM Project," *Music Perception*, zv. 26, št. 3, str. 187–194, 2009.

[31] F. Lerdahl in R. Jackendoff, *A Generative Theory of Tonal Music*. Cambridge: M.I.T. Press, 1983.

[32] H. B. Lincoln, "Uses of the Computer in Music Composition and Research," *Advances in Computers*, zv. 12, str. 73–114, 1972.

[33] A. Marsden, "Schenkerian Analysis by Computer: A Proof of Concept," *Journal of New Music Research*, zv. 39, št. 3, str. 269–289, 2010.

[34] R. Marxer, J. Janer in J. Bonada, "Low-latency instrument separation in polyphonic audio using timbre models," v *Latent Variable Analysis*

and Signal Separation (ur. F. Theis, A. Cichocki, A. Yeredor in M. Zibulevsky), Berlin: Springer Berlin Heidelberg, 2012, str. 314–321.

[35] J. H. McDermott in A. J. Oxenham, "Music perception, pitch, and the auditory system," *Current Opinion in Neurobiology*, zv. 18, št. 4, str. 452–463, 2008.

[36] D. Meredith, "The computational representation of octave equivalence in the Western staff notation system," v *Cambridge Music Processing Colloquium, September 1999*, 1999.

[37] ——, "COSIATEC and SIATECCompress: Pattern Discovery by Geometric Compression," v *Music Information Retrieval Evaluation eXchange*, 2014.

[38] D. Meredith, K. Lemström in G. A. Wiggins, "Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music," *Journal of New Music Research*, zv. 31, št. 4, str. 321–345, 2002.

[39] J. A. Moorer, "On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer," Doktorska disertacija, Stanford University, 1975.

[40] M. Müller in M. Clausen, "Transposition-Invariant Self-Similarity Matrices," v *Proc. of the 8th International Society for Music Information Retrieval Conference*, Vienna, Austria, 2007, str. 47–50.

[41] O. Nieto in M. M. Farbood, "Identifying Polyphonic Patterns From Audio Recordings Using Music Segmentation Techniques," v *Proc. of the 15th International Society for Music Information Retrieval Conference*, Taipei, Taiwan, 2014, str. 411–416.

[42] ——, "Mirex 2014 Entry: Music Segmentation Techniques and Greedy Path Finder Algorithm to Discover Musical Patterns," v *Music Information Retrieval Evaluation eXchange*, 2014.

[43] M. Pesek, A. Leonardis in M. Marolt, "A compositional hierarchical model for music information retrieval," v *Proc. of the 15th International Society for Music Information Retrieval Conference*, Taipei, Taiwan, 2014, str. 131–136.

[44] J. Pickens, "Harmonic Modeling for Polyphonic Music Retrieval," Ph.D. dissertation, University of Massachusetts Amherst, 2004.

[45] C. S. Sapp, "Visual Hierarchical Key Analysis," *Computers in Entertainment*, zv. 4, št. 4, 2005.

[46] H. Schenker, *Harmony*. London: University of Chicago Press, 1954.

[47] W. A. Schloss, "On the Automatic Transcription of Percussive Music – From Acoustic Signal to High-Level Analysis," Doktorska disertacija, Stanford University, 1985.

[48] A. W. Slawson, "Vowel Quality and Musical Timbre as Functions of Spectrum Envelope and Fundamental Frequency," *Journal of the Acoustical Society of America*, zv. 43, št. 1, str. 87–101, 1968.

[49] L. Stewart, T. Overath, J. D. Warren, J. M. Foxton in T. D. Griffiths, "fMRI Evidence for a Cortical Hierarchy of Pitch Pattern Processing," *PLoS ONE*, zv. 3, št. 1, 2008.

[50] G. Velarde, T. Weyde in D. Meredith, "An approach to melodic segmentation and classification based on filtering with the Haar wavelet," *Journal of New Music Research*, zv. 42, št. 4, str. 325–345, 2013.

[51] G. Velarde in D. Meredith, "A Wavelet-based Approach to the Discovery of Themes and Sections in Monophonic Melodies," v *Music Information Retrieval Evaluation eXchange*, 2014.

[52] C. Wang, J. Hsu in S. Dubnov, "Music Pattern Discovery with Variable Markov Oracle: A Unified Approach to Symbolic and Audio Representa-

tions," in *Proc. of the 16th International Society for Music Information Retrieval Conference*, Malaga, Spain, 2015, str. 176–182.

[53] E. Wold, T. Blum, D. Keislar in J. Wheaton, "Content-Based Classification, Search, and Retrieval of Audio," *IEEE MultiMedia*, zv. 3, št. 3, str. 27–36, 1996.

Dodatek A

Primer izhodne datoteke z rezultati

pattern1
occurrence1
37.5000000000, 81.0000000000
37.6250000000, 79.0000000000
37.7500000000, 77.0000000000
37.8750000000, 76.0000000000
occurrence2
134.5000000000, 69.0000000000
134.6250000000, 67.0000000000
134.7500000000, 65.0000000000
134.8750000000, 64.0000000000
occurrence3
162.5000000000, 76.0000000000
162.6250000000, 74.0000000000
162.7500000000, 72.0000000000
162.8750000000, 71.0000000000
pattern2
occurrence1

242.7500000000, 52.0000000000
242.8750000000, 53.0000000000
243.0000000000, 55.0000000000
243.6250000000, 53.0000000000
243.7500000000, 52.0000000000
occurrence2
39.0000000000, 71.0000000000
39.1250000000, 72.0000000000
39.2500000000, 74.0000000000
39.6250000000, 72.0000000000
39.7500000000, 71.0000000000
occurrence3
237.0000000000, 51.0000000000
237.1250000000, 52.0000000000
237.2500000000, 54.0000000000
237.6250000000, 52.0000000000
237.7500000000, 51.0000000000
occurrence4
165.0000000000, 63.0000000000
165.1250000000, 64.0000000000
165.2500000000, 66.0000000000
165.6250000000, 64.0000000000
165.7500000000, 63.0000000000